IBM WebSphere Application Server Network Deployment for Distributed Platforms, Version 8.0

Setting up intermediary services

Note

Before using this information, be sure to read the general information under “Notices” on page 365.

Compilation date: July 15, 2011

Contents

How to send your comments

Your feedback is important in helping to provide the most accurate and highest quality information.

- To send comments on articles in the WebSphere Application Server Information Center
  1. Display the article in your Web browser and scroll to the end of the article.
  2. Click on the **Feedback** link at the bottom of the article, and a separate window containing an e-mail form appears.
  3. Fill out the e-mail form as instructed, and click on **Submit feedback** .
- To send comments on PDF books, you can e-mail your comments to: **wasdoc@us.ibm.com** or fax them to 919-254-5250.

  Be sure to include the document name and number, the WebSphere Application Server version you are using, and, if applicable, the specific page, table, or figure number on which you are commenting.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Changes to serve you more quickly

Print sections directly from the information center navigation

PDF books are provided as a convenience format for easy printing, reading, and offline use. The information center is the official delivery format for IBM WebSphere Application Server documentation. If you use the PDF books primarily for convenient printing, it is now easier to print various parts of the information center as needed, quickly and directly from the information center navigation tree.

To print a section of the information center navigation:

1. Hover your cursor over an entry in the information center navigation until the **Open Quick Menu** icon is displayed beside the entry.
2. Right-click the icon to display a menu for printing or searching your selected section of the navigation tree.
3. If you select **Print this topic and subtopics** from the menu, the selected section is launched in a separate browser window as one HTML file. The HTML file includes each of the topics in the section, with a table of contents at the top.
4. Print the HTML file.

For performance reasons, the number of topics you can print at one time is limited. You are notified if your selection contains too many topics. If the current limit is too restrictive, use the feedback link to suggest a preferable limit. The feedback link is available at the end of most information center pages.

Under construction!

The Information Development Team for IBM WebSphere Application Server is changing its PDF book delivery strategy to respond better to user needs. The intention is to deliver the content to you in PDF format more frequently. During a temporary transition phase, you might experience broken links. During the transition phase, expect the following link behavior:

- Links to Web addresses beginning with http:// work
- Links that refer to specific page numbers within the same PDF book work
- The remaining links will *not* work. You receive an error message when you click them

Thanks for your patience, in the short term, to facilitate the transition to more frequent PDF book updates.

Chapter 1. How do I...Setting up intermediary services

Follow these shortcuts to get started quickly with popular tasks.

When you visit a task in the information center, look for the **IBM Suggests** feature at the bottom of the page. Use it to find available tutorials, demonstrations, presentations, developerWorks articles, IBM Redbooks, support documents, and more.

Create WebSphere profiles.

Setup the proxy server.

Setting up DataPower appliance manager

Use the administrative console to establish communication with local and remote web servers.

Use scripting to establish communication with local and remote web servers.

Configure transport chains.

Configure additional nodes.

Customize the File Synchronization Service.

Provide access to relational databases (JDBC resources) with scripting

Choosing a messaging provider

Provide access to messaging resources (default messaging provider) with scripting

Chapter 2. Implementing a web server plug-in

This topic describes how to implement a web server plug-in. The product works with a web server to route requests for dynamic content, such as servlets, from web applications. The web servers are necessary for directing traffic from browsers to the applications that run on an application server. The web server plug-in uses the XML configuration file to determine whether a request is for an application server.

Before you begin

- See the information about choosing a front end for your WebSphere® Application Server topology." This topic helps you determine whether to set up a web server plug-in, a proxy server, or a secure proxy server to provide session affinity, failover support, and workload balancing for your WebSphere Application Server topology. Install your web server if it is not already installed.

  If you want to use the IBM® HTTP Server that is provided with the product, see the *Installing your application serving environment* PDF. Otherwise, see the installation information that is provided with your web server.
- Verify that your web server is configured to run the operations that are required by web applications, such as GET and POST. Typically, configuring your web server to run these operations involves setting a directive in the web server configuration file. Refer to the web server documentation for instructions.

  If an operation is not enabled when a servlet or JavaServer Pages (JSP) file requiring the operation is accessed, an error message is displayed, such as this one from the IBM HTTP Server:

  ```
  HTTP method POST is not supported by this URL.
  ```
- Make sure the appropriate plug-in file has been installed on your web server and the `configure`*`web_server_name`* script has been run to create and configure the web server definition for this web server.

  If you are using a distributed platform web server, use the web Server Plug-ins Configuration Tool to install the appropriate plug-in file to your web server. Then, run the `configure`*`web_server_name`* script created by the tool to create and configure the web server definition in the WebSphere configuration repository.

If you are making a series of simultaneous changes, such as installing numerous applications, you might want the configuration service disabled until after you make the last change. The web server plug-in configuration service is enabled by default. To disable this service, in the administrative console click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **administration services** > **web server plug-in configuration service**, and then clear the **Enable automated web server configuration processing** option.

gotcha: If your installation uses a firewall, make sure that you configure the web server plug-in to use a port that has been opened. See your security administrator for information about how to obtain an open port.

About this task

The following steps are performed during the plug-in installation process. See the Plug-in Installation Roadmap for additional information.

1. A node is created.

   An unmanaged node is created when the web server is on a different computer from the application server. An unmanaged node is a node that does not have a node agent running on it. Using unmanaged nodes, the product can represent servers that are not application servers within its configuration topology. This representation enables connection information between those servers and application servers to be maintained. The *Using the administrative clients* PDF describes how to create a node.
2. A web server definition is created.

You can also use either the administrative console or use the `ConfigurewebServerDefintion.jacl` script to create a web server definition.

3. An application or modules are mapped to a web server. If an application that you want to use with this web server is already installed, the application is automatically mapped to the web server. If the application is not installed, select this web server during the "Map modules to servers" step of the application installation process.
4. The master repository is updated and saved.

When you configure a plug-in, the configuration file for that plug-in is automatically created. You can change or tune the default settings for the properties in this configuration file. If you change any of the settings, you must regenerate the file before your changes take effect.

Generating or regenerating the configuration file might take a while to complete. After it finishes, all objects in the administrative cell use their newest settings, which the web server can access. If the application server is on the same physical workstation as the web server, the regeneration usually takes about 30 to 60 seconds to complete. The regeneration takes longer if the application server and web server are on different workstations.

The following procedure describes the steps for updating the plug-in configuration file, including configuring for SSL and web server tuning.

Procedure

1. Use the administrative console to change the settings in the plug-in configuration file.

   When setting up your web server plug-in, you must decide whether to have the configuration automatically generated in response to a configuration change. When the web server plug-in configuration service is enabled and any of the following conditions occur, the plug-in configuration file is automatically generated:
   - When the web server is created or saved
   - When an application is installed
   - When an application is uninstalled
   - When the virtual host definition is updated

   gotcha: When the plug-in configuration file is first generated, it does not include admin_host on the list of virtual hosts. The *Installing your application serving environment* PDF describes how to add it to the list.

   You can either use the administrative console, or issue the GenPluginCfg command to regenerate your `plugin-cfg.xml` file.

   Complete the following steps to regenerate your `plugin-cfg.xml` file by using the administrative console:

   a. Select **Servers > Server Types > web Servers >** *web_server_name* **> plug-in properties**.

   b. Select **Automatically generate plug-in configuration file,** or click one or more of the following topics to manually configure the `plugin-cfg.xml` file:

      Note: You must delete the `plugin-cfg.xml` file in the *profile_root*/`config/cells` directory before you complete this task. Otherwise, configuration changes do not persist to the `plugin-cfg.xml` file.
      - Caching
      - Request and response
      - Request routing
      - Custom Properties

      See the topic about web server plug-in configuration properties for information about how to map each property to one of these topics.

gotcha: Do not manually update the `plugin-cfg.xml` file. Any manual updates you make for a web server are overridden whenever the `plugin-cfg.xml` file for that web server is regenerated.

c. Click **OK**.

d. You might have to stop the application server and then start the application server for the web server to locate the `plugin-cfg.xml` file.

2. Install the appropriate GSKIT installation image file on your workstation, if you want to use Secure Sockets Layer (SSL) with this configuration.
3. Tune your web server. See the *Tuning guide* PDF for more information.
4. Propagate the plug-in configuration. The plug-in configuration file, `plugin-cfg.xml`, is automatically propagated to the web server if the web server plug-in configuration service is enabled, and one of the following conditions is true:
   - The web server is a local web server, which means that the web server is located on the same workstation as an application server.
   - The web server is a remote IBM HTTP Server Version 7 that has a running IBM HTTP Server administration server.

   If neither of these conditions are true, you must manually copy the `plugin-cfg.xml` file to the location of the remote web server installation. Copy the `plugin-cfg.xml` file in <*app_server_root*>`/profiles/`<*profilename*>`/config/cells/../../nodes/../servers/`<*webservername*> to the web server host location, which is <*PluginInstallRoot*>`/config/`<*webservername*>`/`.

   Important: If you use the FTP function to copy the file, and the configuration reload fails, check the file permissions on the `plugin-cfg.xml` file, and make sure that they are set to `rw-r--r--`. If the file permissions are not correct, the web server is not able to access the new version of the file, which causes the configuration reload to fail.

   If the file permissions are incorrect, issue the following command to change the file permissions to the appropriate settings:

   ```
   chmod 644 plugin-cfg.xml
   ```

   AIX The AIX® FTP function does not preserve file attributes. Therefore, if you need to manually copy the `plugin-cfg.xml` from an AIX operating system, you might want to use the AIX RCP function instead of the FTP function to copy the file.

   The remote web server installation location is the location you specified when you created the node for this web server.

Results

The configuration is complete. To activate the configuration, stop and restart the web server. If you encounter problems restarting your web server, check the `http_plugin.log` file for information about what portion of the `plugin-cfg.xml` file contains an error. The log file states the line number on which the error occurred, along with other details that might help you diagnose why the web server did not start. You can then use the administrative console to update the `plugin-cfg.xml` file.

If applications are infrequently installed or uninstalled, which is usually the situation in a production environment, or if you can tolerate the performance impact of generating and distributing the plug-in configuration file each time any of the previously listed actions occur, consider enabling the configuration service.

Setting up a local web server

This topic describes how to install the web server and the web server plug-in on the machine where you installed WebSphere Application Server.

Before you begin

If the web server that you are setting up is an IBM HTTP Server, and you plan to manage that web server through a node agent that is running as a nonroot user, you must make sure that you adhere to the following requirements:

- The user ID that you designate as the user ID that owns the IBM HTTP Server directories and files, is the same user ID under which the nonroot node agent is running. You cannot run an IBM HTTP Server as a root user if the node agent that is managing that IBM HTTP Server is running as nonroot node agent because a node agent process that is running as a nonroot user cannot spawn off an IBM HTTP Server that is running as a root user.
- The value you specify for the listener port value must be greater than 1024. An IBM HTTP Server that is running under a nonroot user ID does not start if the port number for its listener port is 1024 or less.

You can ensure that the nonroot node agent and the IBM HTTP Server are using the same user ID if you specify the user ID that you used to install the product as the user ID for the IBM HTTP Server when you install the IBM HTTP Server. However, if, you decide to run the node agent as a nonroot user after you install the IBM HTTP Server and web server plug-in, you can take the following actions to enable both the node agent and the IBM HTTP Server to run as nonroot users:

1. Change the user ID for WebSphere Application Server to a nonroot user ID.
2. Configure the run-as setting for the node agent.
3. Use the administrative console to create a new IBM HTTP Web Server, unless an already defined IBM HTTP Server has the required properties.
4. Change the ownership of the IBM HTTP Server directory and files to the nonroot user ID under which the nonroot node agent is running.

About this task

You can define a locally installed Web server on an unmanaged or managed node. If the web server is defined on an unmanaged node, the administrative functions are handled through the IBM HTTP Server administration server. If the web server is defined on a managed node, the administrative functions of the web server are handled through the WebSphere Application Server node agent, which is beneficial.

Important: Web servers that are *not* provided with the WebSphere Application Server product do not provide an administration server. Web servers that do not provide an administration server must reside on a managed node to facilitate plug-in administration functions and generation and propagation of the `plugin-cfg.xml` file.

The following steps create a web server definition in the default profile.

Procedure

1. Install IBM Installation Manager.
2. Install your WebSphere Application Server product.
3. Install IBM HTTP Server or another supported web server.
4. Install the web server plug-ins.
5. Install the WebSphere Customization Toolbox.
6. Configure the web server plug-in using the Web Server Plug-ins Configuration Tool.

   The web server definition is automatically created and configured.
7. Complete the setup by creating the web server definition using the WebSphere Application Server administrative console, or run the plug-in configuration script. The creation of this object is exclusive of the web server installation.

   You must create an application server profile or a custom profile and federate the node before you can use the administrative console of the deployment manager to create a web server definition. The same is true for running the configuration script that the Web Server Plug-ins Configuration Tool created. You must assign the web server to a managed node when you create it. The managed node must exist before running the Web Server Plug-ins Configuration Tool. Otherwise, the installation is considered a remote installation.

Select one of the following options:

- **Using the administrative console.**

  Create a web server definition on an existing application server or unmanaged node:

  a. Click **Servers > Server Types > Web servers > New** and use the **Create new web server definition** tool to create the web server definition.
  b. Select the appropriate node.
  c. Select a template. Select a system template or a user-defined template for the web server that you want to create.
  d. Enter the web server properties:
     - Type: The web server vendor type
     - Port: The existing web server port (default: 80)
     - Installation path: The web server installation path. This field is required for IBM HTTP Server only.
     - Service name (Windows operating systems): The Windows operating system service name of the web server. The default is `IBMHTTPServer7.0`.
     - Use secure protocol: Use the HTTPS protocol to communicate with the web server. The default is `HTTP`.
     - Plug-in installation location: The directory path in which the plug-in is installed.
  e. Confirm the creation of the new web server, and click **Finish**.

  After creating the web server, complete the following steps to verify that the `plugin-key.kdb` file is generated and to configure the web server plug-in with SSL:

  a. Click **Security > SSL certificate and key management**.
  b. Under Configuration settings, click **Manage endpoint security configurations**.
  c. Under Inbound or Outbound, expand ***cell_name*** **> nodes >** ***Web_server_node_name*** **> servers** and click ***server_name***.
  d. Under Related Items, click **Key stores and certificates**. The administrative console displays the CMSKeyStore configuration with the path to the `plugin-key.kdb` file.
  e. Export the default certificate from `key.p12`, and add it as a signer certificate to the `plugin-key.kdb`.

- **Running the plug-in configuration script.**

  If you install the plug-in, save the plug-in configuration script to run after you create a managed node, otherwise an error occurs. Wait until the script runs successfully and creates the web server definition on the managed node and node synchronization occurs before starting the web server.

  Adding the node starts the node agent process. If the node agent is not running, start the node.

  Tip: If you want the web server to handle requests for an application for multiple managed nodes, install the application on each managed node and on the web server definition. The script already contains all of the information that you must gather when using the administrative console option.

  See the *Using the administrative clients* PDF for more information.

What to do next

You can configure non-IBM HTTP Server Web servers as a remote web server on unmanaged nodes, or as a local Web server on managed nodes. For a non-IBM HTTP Server web server on a managed node, the following functions are supported:

- Generation of the plug-in configuration, based on WebSphere Application Server repository changes.
- Propagation of the `plugin-cfg.xml` file, based on using node synchronization with the WebSphere Application Server node. Node synchronization is necessary in order to propagate configuration changes to the affected node or nodes.

  The `plugin-cfg.xml` file is propagated to the application server node repository tree from the deployment manager repository.

  Important: The `plugin-cfg.xml` file is propagated to the application server node repository tree. This is not the default `plugin-cfg.xml` file installation location. Changes may have to be made to non-IBM HTTP Server web server configuration files to update the location of the `plugin-cfg.xml` file that is read by the plug-in module.

  For example, Internet Information Services (IIS) has a file name called `plugin-cfg.loc`, which is read by the IIS plug-in modules to determine the location of the `plugin-cfg.xml` file. The `plugin-cfg.loc` file has to be updated to reflect the `plugin-cfg.xml` file location in the application server node repository.

  Other non-IBM HTTP Server web servers have different methods to specify the location of the `plugin-cfg.xml` file for the plug-in module. However, in order for propagation to work, update the location to reflect the location in the application server node repository.

The following functions are not supported on a managed node for a non-IBM web server.
- Starting and stopping the web server.
- Viewing and editing the configuration file.
- Viewing the web server logs.

For a non-IBM HTTP Server web server on an unmanaged node, you can generate plug-in configuration, based on WebSphere Application Server repository changes. The following functions are not supported on an unmanaged node for a non-IBM HTTP Server web server:
- Starting and stopping the web server.
- Viewing and editing the configuration file.
- Viewing the web server logs.
- Propagation of the web server `plugin-cfg.xml` file.

Setting up a remote web server

You can create a web server definition in the administrative console when the web server and the web server plug-in for WebSphere Application Server are on the same machine and the application server is on a different machine. This allows you to run an application server on one platform and a web server on another platform.

Before you begin

With a remote web server installation, WebSphere Application Server can facilitate plug-in administration functions and generation and propagation of the `plugin-cfg.xml` file for IBM HTTP Server for WebSphere Application Server, but not for other web servers.

Web servers that are not IBM HTTP Server for WebSphere Application Server must reside on the same machine as the WebSphere Application Server (as a managed node) to facilitate plug-in administration functions and generation and propagation of the `plugin-cfg.xml` file.

About this task

You can choose a remote web server installation if you want the web server on the outside of a firewall and WebSphere Application Server on the inside of a firewall. You can create a remote web server on an unmanaged node. Unmanaged nodes are nodes without node agents. Because there is no WebSphere

Application Server or node agent on the machine that the node represents, there is no way to administer a web server on that unmanaged node unless the web server is IBM HTTP Server for WebSphere Application Server. With IBM HTTP Server, there is an administration server that will facilitate administrative requests such as start and stop, view logs, and view and edit the `httpd.conf` file.

Important: The administration server is not provided with IBM HTTP Server for WebSphere Application Server which runs on z/OS® platforms. So, administration using the administrative console is not supported for IBM HTTP Server for z/OS on an unmanaged node.

The following steps will create a web server definition in the default profile. This procedure does not apply when setting up a remote web server for an i5/OS® web server. For information about setting up an i5/OS web server, see the topic entitled *Selecting a web server topology diagram and roadmap*.

Procedure

1. Install IBM Installation Manager.
2. Install your WebSphere Application Server product.
3. Install IBM HTTP Server or another supported web server.
4. Install the web server plug-ins.
5. Install the WebSphere Customization Toolbox.
6. Configure the web server plug-in using the Web Server Plug-ins Configuration Tool.
7. Complete the setup by creating the web server definition.

   You can use the WebSphere Application Server administrative console or run the plug-in configuration script:

   - **Using the administrative console:**
     a. Click **System Administration > Nodes > Add Node** to create an unmanaged node in which to define a web server in the topology.
     b. Click **Servers > Server Types > Web servers > New** to launch the **Create new web server definition** tool. You will create the new web server definition using this tool. The values are as follows:
        1) Select appropriate node
        2) Enter web server properties:
           – **Type**: The web server vendor type.
           – **Port**: The existing web server port. The default is 80.
           – **Installation Path**: The web server installation path. This field is required field for IBM HTTP Server only.
           – **WINDOWS Service Name**: The Windows operating system service name of the web server. The default is `IBMHTTPServer7.0`.
           – **Use secure protocol**: Use the HTTPS protocol to communicate with the web server. The default is `HTTP`.
           – **Plug-in installation location**: The directory path where the plug-in is installed.
           – **Application mapping to the web server**: Whether you want to create a mapping to existing applications that are currently deployed to the web server. Select `ALL` if you want the mapping created; select `None` if you do not want the mapping created.

             CAUTION:
             If you have enterprise applications in different security domains when you create a web server, the Key Database (KDB) files for your security configuration might not be created if you have Application mapping to the web server set to `All`. To resolve this problem, create the web server with Application mapping to the web server set to `None`. Then map the applications to the web server. All the KDB files for the web server are then created.

3) Enter the remote web server properties. The properties for the IBM HTTP Server administration server follow:
   - **Port**: The administration server port. The default is `8008`.
   - **User ID**: The user ID that is created using the htpasswd script.
   - **Password**: The password that corresponds to the user ID created with the `htpasswd` script.
   - **Use secure protocol**: Use the HTTPS protocol to communicate with the administration server. The default is `HTTP`.

4) Select a web server template. Select a system template or a user-defined template for the web server you want to create.

5) Confirmation of web server creation.

- **Run the plug-in configuration script.**

8. **For AIX, HP-UX, Linux or Solaris operating system**: On the remote web server, run the `setupadm` script. The administration server requires read and write access to configuration files and authentication files to perform web server configuration data administration. You can find the `setupadm` script in the `<`*`IHS_install_root`*`>/bin` directory. The administration server has to execute `adminctl restart` as root to perform successful restarts of IBM HTTP Server. In addition to the web server files, you must manually change the permissions to the targeted plug-in configuration files.

   The `setupadm` script prompts you for the following input:
   - User ID - The user ID that you use to log on to the administration server. The script creates this user ID.
   - Group name - The administration server accesses the configuration files and authentication files through group file permissions. The script creates the specified group through this script.
   - Directory - The directory where you can find configuration files and authentication files.
   - File name - The following file groups and file permissions change:
     - Single file name
     - File name with wildcard
     - All (default) - All of the files in the specific directory
   - Processing - The `setupadm` script changes the group and file permissions of the configuration files and authentication files.

   In addition to the web server files, you must change the permissions to the targeted plug-in configuration files. See the topic on setting permissions manually for instructions.

9. **For AIX, HP-UX, Linux, Solaris, or Windows operating system**: On the remote web server, run the `htpasswd` script. The administration server is installed with authentication enabled and a blank `admin.passwd` password file . The administration server will not accept a connection without a valid user ID and password. This is done to protect the IBM HTTP Server configuration file from unauthorized access.

   Launch the **htpasswd** utility that is shipped with the administration server. This utility creates and updates the files used to store user names and password for basic authentication. Locate **htpasswd** in the `bin` directory.
   - On Windows operating systems: `htpasswd -cm <`*`install_dir`*`>\conf\admin.passwd [`*`login name`*`]`
   - On AIX, HP-UX, Linux, and Solaris platforms: `./htpasswd -cm <`*`install_dir`*`>/conf/admin.passwd [`*`login name`*`]`

   where `<`*`install_dir`*`>` is the IBM HTTP Server installation directory and *[login name]* is the user ID that you use to log into the administration server. The [login name] is the user ID that you entered in the user ID field for the remote web server properties in the administrative console.

10. Start IBM HTTP Server. Refer to the topic on starting and stopping the IBM HTTP Server Administration server for instructions.

What to do next

For a non-IBM HTTP Server web server on an unmanaged node, you can generate a plug-in configuration, based on WebSphere Application server repository changes. However, the following functions are not supported on an unmanaged node for a non-IBM HTTP Server web server:

- Starting and stopping the web server.
- Viewing and editing the web server configuration file.
- Viewing the web server logs.
- Propagation of the web server `plugin-cfg.xml` file.

You can configure non-IBM HTTP Server web servers as a local web server on a managed node. For a non-IBM HTTP Server web server on a managed node, the following functions are supported:

- Generation of the plug-in configuration, based on WebSphere Application Server repository changes.
- Propagation of the `plugin-cfg.xml` file, based on using node synchronization with the WebSphere Application Server node. Node synchronization is necessary in order to propagate configuration changes to the affected node or nodes.

  Note: When WebSphere Application Server is installed using a stand-alone profile on one machine and IBM HTTP Server is installed on a different machine as root user using the administrative server, to ensure that propagation functions correctly, the root user must manually change the permissions of the plugin-cfg.xml file to the nonroot user running IBM HTTP Server from the administrative server. The username and group needed to start the administrative server are located in the HTTPServer/config/admin.conf file.

  The `plugin-cfg.xml` file is propagated to the application server node repository tree from the deployment manager repository.

  Important: The `plugin-cfg.xml` file is propagated to the application server node repository tree. This is not the default `plugin-cfg.xml` file installation location. Changes may have to be made to non-IBM HTTP Server web server configuration files to update the location of the `plugin-cfg.xml` file that is read by the plug-in module.

  For example, Internet Information Services (IIS) has a file name called `plugin-cfg.loc`, which is read by the IIS plug-in modules to determine the location of the `plugin-cfg.xml` file. The `plugin-cfg.loc` file has to be updated to reflect the `plugin-cfg.xml` file location in the application server node repository.

  Other non-IBM HTTP Server web servers have different methods to specify the location of the `plugin-cfg.xml` file for the plug-in module. However, in order for propagation to work, update the location to reflect the location in the application server node repository.

For a non-IBM HTTP Server Web server that is configured as a local web server on a managed node, the following functions are not supported:

- Starting and stopping the web server.
- Viewing and editing the configuration file.
- Viewing the web server logs.

Installing IBM HTTP Server

This topic describes how to install IBM HTTP Server.

Before you begin

Install the IBM HTTP Server product and its plug-in, or install a plug-in for another supported web server to enable the web server to work with WebSphere Application Server.

To use a web server other than IBM HTTP Server, install and configure the web server before or after installing the WebSphere Application Server product but before installing the Web Server Plug-ins for IBM WebSphere Application Server.

About this task

Refer to the information center for IBM HTTP Server for detailed information on installation steps, configuring the web server for SSL, the Fast Response Cache Accelerator, or Apache directives.

What to do next

After installing the web server and the application server, install and configure the appropriate web server plug-in for a supported, installed web server. No further configuration is required for most web servers.

The Web Server Plug-ins Configuration Tool configures supported web servers. You can also manually configure supported Web servers for WebSphere Application Server as described in “Editing web server configuration files.”

The Global Security Kit (GSKit) installation image files for the WebSphere Web server plug-ins are packaged on the CD with the web server plug-in files. If you want to use GSKIT, you can download the appropriate GSKIT file to the workstation on which your web server is running.

Editing web server configuration files

Edit web server configuration files to configure a Web server.

Before you begin

The Web Server Plug-ins Configuration Tool configures supported web servers. Use this topic to understand how the tool configures the web server configuration files.

If you must change a configuration for some reason, you can run the Web Server Plug-ins Configuration Tool again to re-configure the web server or you can edit the files.

To support a nondefault profile, edit the configuration to point the web server to the correct location of the plug-in configuration file (`plugin-cfg.xml`). In this case, you would change the profile path from the default profile to the secondary profile.

You must determine whether your web server application uses a 32-bit or 64-bit architecture. If your machine supports a 64-bit architecture, you can use either a 32-bit or a 64-bit web server application. The following examples show how to determine which the architecture for your web server application:

- For Apache-based web servers, including IBM HTTP Server, you can determine the architecture using the following techniques:
  - AIX HP-UX Linux Solaris Run the command:

    ```
    apachectl -V | grep Architecture
    ```

    This command displays the Architecture value for the web server, which will be either 32-bit or 64-bit.
  - Windows Run the following command from the `*IBM_HTTP_Server_install_root*\bin` directory:

    ```
    apache -V
    ```

    Look for the value of the `Architecture:` line in the output. This value is either 32-bit or 64-bit.
- For other, non-Apache based web servers, use the following techniques. These techniques also work for Apache-based web servers.
  - AIX HP-UX Linux Solaris Run the file command against the primary web server application executable file to display its type. For example, run the following command on an AIX system:

`file /usr/bin/httpd`

This command results in the following output:

- For a 64-bit web server: `httpd: 64-bit XCOFF executable ...`
- For a 32-bit web server: `httpd: executable (RISC System/6000)...`

In general, if the output of the file command does not indicate `64` in the output, then the application is 32-bit application.

– Windows Open the Microsoft Windows Task Manager when the server is running and locate the server in the process list. A 32-bit application has *32 after its name.

If the web server application uses a 64-bit architecture, the plug-in library should be loaded from the *plugin_root*/bin/64bit directory. If the web server application uses a 32-bit architecture, the plug-in library should be loaded from the *plugin_root*/bin/32bits directory. If the *plugin_root*/bin/64bits directory does not exist, you must use the 32-bit architecture.

About this task

This task points you to information on editing web server configuration files. Select a link appropriate for your web server.

Procedure

- Configure Apache HTTP Server 2.2. See "Configuring Apache HTTP Server V2.2" on page 16.
- Configure Domino® Domino Web Server Version 5 and Version 6.x. See "Configuring Lotus Domino" on page 18.
- Configure IBM HTTP Server powered by Apache 2.x. See "Configuring IBM HTTP Server powered by Apache 2.x" on page 21.
- Configure IBM HTTP Server Version 6.x. See "Configuring IBM HTTP Server Version 6.x" on page 23.
- Configure IBM HTTP Server Version 7.0. See "Configuring IBM HTTP Server Version 7.0" on page 25.
- Configure IBM HTTP Server Version 78.0. See "Configuring IBM HTTP Server Version 8.0" on page 27.
- Configure Microsoft Internet Information Services (IIS). See "Configuring Microsoft Internet Information Services (IIS)" on page 29.
- Configure Sun Java System Web Server (formerly Sun ONE and iPlanet). See "Configuring the Sun Java System Web Server" on page 34.

Results

You can use the Web Server Plug-ins Configuration Tool to configure supported web servers. You can also configure a web servers by editing its configuration.

Configuring Apache HTTP Server V2.0

This topic describes how to change configuration settings for Apache HTTP Server Version 2.0.

Before you begin

After you install the web server plug-ins, you can use the Web Server Plug-ins Configuration Tool to configure a web server plug-in.

This topic describes how to configure the Apache HTTP Server Version 2.0 Web Server. Other procedures in "Editing web server configuration files" on page 12 describe configuring other supported web servers.

gotcha:

- If you are using an Apache HTTP Server that supports 64-bit addressing, you must use the 64-bit CD provided with the WebSphere Application Server product to install the Apache Web Server plug-in binaries. If you use the 32-bit CD, you will receive an error message indicating that the plug-in binaries did not load.
- If you are using an Apache HTTP Server that supports 32-bit addressing, you must use the 32-bit CD provided with the WebSphere Application Server product to install the Apache Web Server plug-in binaries. If you use the 64-bit CD, you will receive an error message indicating that the plug-in binaries did not load.

A sample error message follows:

```
httpd: Syntax error on line XXX of /home/apache/conf/httpd.conf: Cannot
load /home/apache/Plugins/mod_was_ap20_http.sl into server: Invalid argument
```

The plug-in was tested with the threaded worker multi-processing module (MPM) on all platforms except Windows. The plug-in was tested with the default threaded MPM on Windows.

The plug-in works with the Apache 2 prefork MPM but works best with the worker MPM. The plug-in maintains connection pools to backend WebSphere Application Servers and uses in-memory caching. These plug-in functions perform most efficiently when Apache 2.0 is configured to use a single child process with the ThreadsPerChild value equal to the MaxClients value. The plug-in can be used with the prefork MPM or the worker MPM that is configured with multiple child processes, but at reduced efficiency.

Compatibility Statement The plug-in works with versions of the Apache HTTP Server that claim full binary compatibility with Apache 2.0.47 and later, which are built with compilers and compiler options that are compatible with those used to build the plug-in.

About this task

Perform the step that configures Apache 2.0 for your operating system.

Examples and messages are shown on more than one line for clarity. Place each directive in a web server configuration file on one line.

Local file path means a file path to the `plugin-cfg.xml` file on an application server that is on the same machine as the web server. Remote file path means the file path to the `plugin-cfg.xml` file when the application server is on a remote machine.

The Web Server Plug-ins Configuration Tool installs a dummy `plugin-cfg.xml` file during installation, but this file requires periodic propagation from the real file on the application server machine.

The *node_name* in the following application server local file paths is *web_server_name*_`node` for a standalone application server or *managed_node_name* for a managed node.

The name of the web server definition in the following steps is webserver1.

Procedure

- AIX Configure entries in the `httpd.conf` file.

  Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

  ```
  LoadModule
       was_ap20_module /usr/IBM/WebSphere/Plugins/bin/mod_was_ap20_http.so
  ```

  Local distributed example:

  ```
  WebSpherePluginConfig
       profile_root/config/cells/
          dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
  ```

Example:

```
WebSpherePluginConfig
     /usr/IBM/WebSphere/Plugins/config/webserver1/plugin-cfg.xml
```

- Linux Solaris Configure entries in the `httpd.conf` file.

  Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

  ```
  LoadModule
       was_ap20_module /opt/IBM/WebSphere/Plugins/bin/mod_was_ap20_http.so
  ```

  Local distributed example:

  ```
  WebSpherePluginConfig
       profile_root/config/cells/
          dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
  ```

  Example:

  ```
  WebSpherePluginConfig
       /opt/IBM/WebSphere/Plugins/config/webserver1/plugin-cfg.xml
  ```

  Solaris On the Solaris SPARC 64-bit platform, the Web Server Plug-ins Configuration Tool installs both 32-bit and 64-bit versions of the plug-in for Apache 2.0; however, it configures the web server to use the 32-bit plug-in only. If the web server is 64-bit, you need to configure the LoadModule directive in the `httpd.conf` file to use the 64-bit plug-in as follows:

  ```
  LoadModule
       was_ap20_module /usr/IBM/WebSphere/Plugins/bin/64bits/mod_was_ap20_http.so
  ```

- HP-UX Configure entries in the `httpd.conf` file.

  Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

  ```
  LoadModule
       was_ap20_module /opt/IBM/WebSphere/Plugins/bin/mod_was_ap20_http.sl
  ```

  Local distributed example:

  ```
  WebSpherePluginConfig
       profile_root/config/cells/
          dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
  ```

  Example:

  ```
  WebSpherePluginConfig
       /opt/IBM/WebSphere/Plugins/config/webserver1/plugin-cfg.xml
  ```

- Windows Configure entries in the `httpd.conf` file.

  Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

  ```
  LoadModule was_ap20_module
       drive:\IBM\WebSphere\Plugins\bin\mod_was_ap20_http.dll
  ```

  Local distributed example:

  ```
  WebSpherePluginConfig
       profile_root\config\cells\
          dmgrcell\nodes\managednode\servers\webserver1\plugin-cfg.xml
  ```

  Example:

  ```
  WebSpherePluginConfig
       C:\IBM\WebSphere\Plugins\config\webserver1\plugin-cfg.xml
  ```

Results

This procedure results in re-configuring the Apache 2.0 Web Server.

What to do next

The mod_was_ap20_http plug-in module requires the GSKIT Secure Sockets Layer (SSL) encryption library if the plug-in is configured to support encrypted connections to back-end WebSphere Application Servers.

Installing the web server plug-ins installs the GSKIT SSL encryption library at the required level if it is not installed. If you manually copy the plug-in to a new machine, you might not have the required GSKIT libraries for encrypting backend connections.

After configuring a web server, you can install applications on it. See the Applications section of the information center for more information.

Tip: To unconfigure a web server, reverse the manual steps and remove what was manually added in this procedure.

Configuring Apache HTTP Server V2.2

This topic describes how to change configuration settings for Apache HTTP Server Version 2.2.

Before you begin

Install Apache 2.2 and the latest version of the web server plug-ins.

Apache HTTP Server v2.2 is different from IBM HTTP Server (powered by Apache). Apache HTTP Server is not supported on IBM i.

After you install the web server plug-ins, you can use the Web Server Plug-ins Configuration Tool to configure a web server plug-in.

This topic describes how to configure the Apache HTTP Server Version 2.2 Web server. Other procedures in “Editing web server configuration files” on page 12 describe configuring other supported web servers.

gotcha:

- If you are using an Apache HTTP Server that supports 64-bit addressing, you must use the 64-bit CD provided with the WebSphere Application Server product to install the Apache Web Server plug-in binaries. If you use the 32-bit CD, you will receive an error message indicating that the plug-in binaries did not load.
- If you are using an Apache HTTP Server that supports 32-bit addressing, you must use the 32-bit CD provided with the WebSphere Application Server product to install the Apache Web Server plug-in binaries. If you use the 64-bit CD, you will receive an error message indicating that the plug-in binaries did not load.

A sample error message follows:

```
httpd: Syntax error on line XXX of /home/apache/conf/httpd.conf: Cannot
load /home/apache/Plugins/mod_was_ap22_http.sl into server: Invalid argument
```

The plug-in was tested with the threaded worker multi-processing module (MPM) on all platforms except Windows. The plug-in was tested with the default threaded MPM on Windows.

The plug-in works with the Apache 2.2 prefork MPM but works best with the worker MPM. The plug-in maintains connection pools to backend WebSphere Application Servers and uses in-memory caching. These plug-in functions perform most efficiently when Apache is configured to use a single child process with the ThreadsPerChild value equal to the MaxClients value. The plug-in can be used with the prefork MPM or the worker MPM that is configured with multiple child processes, but at reduced efficiency.

Compatibility Statement The plug-in works with versions of the Apache HTTP Server that claim full binary compatibility with Apache 2.0.47 and later, which are built with compilers and compiler options that are compatible with those used to build the plug-in.

About this task

Perform the step that configures Apache 2.2 for your operating system.

Examples and messages are shown on more than one line for clarity. Place each directive in a web server configuration file on one line.

Local file path means a file path to the `plugin-cfg.xml` file on an application server that is on the same machine as the web server. Remote file path means the file path to the `plugin-cfg.xml` file when the application server is on a remote machine.

The Web Server Plug-ins Configuration Tool installs a dummy `plugin-cfg.xml` file during installation, but this file requires periodic propagation from the real file on the application server machine.

The *node_name* in the following application server local file paths is *web_server_name*`_node` for a standalone application server or *managed_node_name* for a managed node.

The name of the web server definition in the following steps is webserver1.

Procedure

- AIX Configure entries in the `httpd.conf` file.

  Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

  ```
  LoadModule
       was_ap22_module /usr/IBM/WebSphere/Plugins/bin/mod_was_ap22_http.so
  ```

  Local distributed example:

  ```
  WebSpherePluginConfig
       profile_root/config/cells/
          dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
  ```

- Linux Solaris Configure entries in the `httpd.conf` file.

  Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

  ```
  LoadModule
       was_ap22_module /opt/IBM/WebSphere/Plugins/bin/mod_was_ap22_http.so
  ```

  Local distributed example:

  ```
  WebSpherePluginConfig
       profile_root/config/cells/
          dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
  ```

  Solaris On the Solaris SPARC 64-bit platform, the Web Server Plug-ins Configuration Tool installs both 32-bit and 64-bit versions of the plug-in for Apache 2.2, however it configures the web server to use the 32-bit plug-in only. If the web server is 64-bit, you need to configure the LoadModule directive in the `httpd.conf` file to use the 64-bit plug-in as follows:

  ```
  LoadModule
       was_ap22_module /usr/IBM/WebSphere/Plugins/bin/64bits/mod_was_ap22_http.so
  ```

- HP-UX Configure entries in the `httpd.conf` file.

  Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

  ```
  LoadModule
       was_ap22_module /opt/IBM/WebSphere/Plugins/bin/mod_was_ap22_http.sl
  ```

  Local distributed example:

```
WebSpherePluginConfig
     profile_root/config/cells/
        dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
```

- Windows Configure entries in the `httpd.conf` file.

  Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

  ```
  LoadModule was_ap22_module
       drive:\IBM\WebSphere\Plugins\bin\mod_was_ap22_http.dll
  ```

  Local distributed example:

  ```
  WebSpherePluginConfig
       profile_root\config\cells\
          dmgrcell\nodes\managednode\servers\webserver1\plugin-cfg.xml
  ```

Results

The Apache 2.2 Web Server is re-configured.

What to do next

The native GSKIT Secure Sockets Layer (SSL) encryption library is used.

Installing the web server plug-ins installs the GSKIT SSL encryption library at the required level if it is not installed. If you manually copy the plug-in to a new machine, you might not have the required GSKIT libraries for encrypting back-end connections.

After configuring a web server, you can install applications on it. See the Applications section of the information center for more information.

Tip: To unconfigure a web server, reverse the manual steps and remove what was manually added in this procedure.

Configuring Lotus Domino

This task describes how to change configuration settings for Lotus® Domino.

Before you begin

The Web Server Plug-ins Configuration Tool configures the web server plug-in.

Other procedures in "Editing web server configuration files" on page 12 describe configuring other supported web servers.

About this task

Use the following procedure to enable the web server plug-in to work with Lotus Domino. Ensure that you install the plug-in as root. Domino can only be installed as root, and the configuration files belong to root.

: If you install the plug-in as local and the WebSphere Application Server as nonroot, then web server management on WebSphere Application Server, such as generation, propagation, deletion of web server definitions and more, is unavailable because the `plugin-cfg.xml` file must be installed as root.

Procedure

1. Start the Domino server.
2. Access the `names.nsf` file using your web browser (for example, `http://tarheels2.raleigh.ibm.com/names.nsf`). The browser prompts you for a password.
3. Give the administrator name and password.

4. Click **Configuration** on the left side of the page.
5. Click **Servers** on the left side of the page.
6. Click **All Server Documents** on the left side of the page.
7. Click the server that you intend to have work with the product
8. Click **Edit Server** on the top-left of the center window.
9. Click **Internet Protocols** in the middle of the page.
10. Add the path to the Domino plug-in under the DSAPI Section in the middle-right of the page.

    If specifications for filter files for the Domino Server Application Programming Interface (DSAPI) exist, use a space to delimit the web server plug-in for WebSphere Application Server.
11. Click **Save and Close** in the upper left of the center window.
12. Define the location of the `plugin-cfg.xml` configuration file.

    The location varies depending on how you have configured your system. If the web server and the application server are on separate machines, you have a remote installation.

    If the two servers are on the same machine, you have a local installation.

    If the two servers are on the same machine and the application server node or the custom node is federated, you have a local distributed installation.

    In the following examples, webserver1 is the web server definition name. AIX HP-UX Linux Solaris

    AIX HP-UX Linux Solaris

Table 1. Setting the path to the plug-in configuration file. Set the WAS_PLUGIN_CONFIG_FILE environment variable to the location of the plug-in configuration file using one of the paths in this table.

If the type of installation is:	Then use this command to set the environment variable:
Remote	`WAS_PLUGIN_CONFIG_FILE=/opt/IBM/WebSphere/Plugins/config/webserver1/plugin-cfg.xml`
Local standalone	`WAS_PLUGIN_CONFIG_FILE=`*`profile_root`*`/config/cells/sa_cell/nodes/webserver1_node/servers/webserver1/plugin-cfg.xml`
Local distributed	`WAS_PLUGIN_CONFIG_FILE=`*`profile_root`*`/config/cells/dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml`

The `setupPluginCfg.sh` file is created in two places:

- The *plugins_root*/`bin` directory
- The *lotus_root*/`notesdata` directory

You can run the script from either location to set the WAS_PLUGIN_CONFIG_FILE environment variable. However, if you are re-configuring the web server, you might want to set the path yourself by setting the value of the environment variable with a path from the preceding table.

The `setupPluginCfg.sh` script sets the file path value to the file path that the Web Server Plug-ins Configuration Tool configured originally. If you are reconfiguring the web server to change the original file path, do not use this script. Windows

Windows

Table 2. Setting the path to the plug-in configuration file. Add the appropriate statement to your lotus_domino_root\ `notes.ini` file.

If the type of installation is:	Then use this command to set the WebSpherePluginCfg variable:
Remote	`WebSpherePluginCfg=C:\Program Files\IBM\WebSphere\Plugins\config\webserver1\plugin-cfg.xml`
Local standalone	`WebSpherePluginCfg=`*`profile_root`*`\config\cells\sa_cell\nodes\webserver1_node\servers\webserver1\plugin-cfg.xml`

Table 2. Setting the path to the plug-in configuration file (continued). Add the appropriate statement to your lotus_domino_root\`notes.ini` file.

If the type of installation is:	Then use this command to set the WebSpherePluginCfg variable:
Local distributed	`WebSpherePluginCfg=`*`profile_root`*`\config\cells\dmgrcell\nodes\managednode\servers\webserver1\plugin-cfg.xml`

Do not delimit any of the following file paths with quotation marks unless there is a space in the file path. Otherwise, the `plugin-cfg.xml` file might not load correctly.

13. If you are configuring a 64-bit version of the Domino Version 7 or higher web server, modify the notes.ini file to point to the *`plugin_root`*`\bin\64bit/domino5.dll` file.
14. Restart the Domino server. When the server starts, information like the following example is displayed:

```
01/21/2005 01:21:51 PM  JVM: Java virtual machine initialized
WebSphere Application Server DSAPI filter loaded
01/21/2005 01:21:52 PM  HTTP Web Server started
```

Results

This procedure results in reconfiguring Version 6.x of Lotus Domino.

What to do next

After configuring a web server, you can install applications on it. See the Applications section of the information center for more information.

For more information about configuring Lotus Domino to work with WebSphere Application Server, search the Lotus Support Services website at http://www-3.ibm.com/software/lotus/support/. Enter the search term `WebSphere` in the keyword search field.

Tip: To unconfigure a web server, reverse the manual steps and remove what was manually added in this procedure.

Lotus Domino file locations and troubleshooting tips

Lotus Domino Server is one of the Web servers that WebSphere Application Server supports. This topic describes configuration file locations and provides other tips related to using Lotus Domino.

Lotus Domino Server file locations

Lotus Domino server file locations are:

- AIX
  - `/usr/lotus/notes/50091/ibmpow/Notes.jar`
  - `/usr/notesdata/names.nsf`
- Solaris
  - `/opt/lotus/notes/5080/sunspa/Notes.jar`
  - `/opt/notesdata/names.nsf`
- Windows
  - `c:\Program Files\lotus\notes\Notes.jar`
  - `c:\Program Files\lotus\notes\data\names.nsf`

Solaris

The Domino Server plug-in might fail to configure on a Solaris platform

If this occurs during installation, a `dsapi_stderr.txt` file is created in the logs directory and you get the following error messages:

```
lotus.notes.NotesException: Could not load dll for system name SUNOS
       at lotus.notes.NotesThread.load(NotesThread.java:210)
       at lotus.notes.NotesThread.<clinit>(NotesThread.java:24)
java.lang.UnsatisfiedLinkError: NnotesInitThread
       at lotus.notes.NotesThread.NnotesInitThread(Native Method)
       at lotus.notes.NotesThread.initThread(NotesThread.java:99)
       at lotus.notes.NotesThread.run(NotesThread.java:133)
```

You can configure the WebSphere Application Server or Domino Server plug-in manually using the Domino Server Web Administration tool. Use the following procedures:

1. Start the Domino Server.
2. Enter the URL for the Domino Server Web Administration site using a browser. For example, `http://`*`host_name`*`/names.nsf`. Enter the administrator user name and password.
3. Double-click **Server-Servers**.
4. Double-click **WebServer** to configure.
5. Double-click **Edit Server**.
6. Double-click **Internet Protocol**.
7. Add the WebSphere Application Server DSAPI plug-in to the **DSAPI** field. For example, *app_server_root*`/bin/libdomino5_http.so`

   If there are already DSAPI filter files specified, use a space to delimit the WebSphere Application Server plug-in file.
8. Double-click **Save and Close**.
9. Restart the Domino Server.

AIX HP-UX Linux Solaris

Avoiding a DSAPI filter-loading error when the Lotus Domino Server starts

On operating systems such as AIX or Linux, if the Lotus Domino Web Server starts using a nonroot user, you are likely to generate a DSAPI filter-loading error when the Lotus Domino Server starts:

```
Error loading DSAPI filter.

Filter not loaded: app_server_root/bin/libdomino6_http.a
```

Manually change the WebSphere Application Server `bin` directory permissions from `750` to `755` to run Lotus Domino Server as a nonroot user and not generate the error.

You must also change permissions on the WebSphere Application Server `logs` directory to `777` to allow Lotus Domino Server to write to the log. However, this change poses a security risk.

If the Lotus Domino Server is started as root, the problem does not occur.

Configuring IBM HTTP Server powered by Apache 2.x

This topic describes how to change configuration settings for IBM HTTP Server powered by Apache 2.x.

Before you begin

After you install the web server plug-ins, you can use the Web Server Plug-ins Configuration Tool to configure a web server plug-in.

This topic describes how to configure the IBM HTTP Server. Other procedures in “Editing web server configuration files” on page 12 describe configuring other supported web servers.

About this task

Perform the step that configures IBM HTTP Server version 2.2 for your operating system. IBM HTTP Server powered by Apache 2.2 is supported on IBM i Versions 6.1 and 7.1.

Examples and messages are shown on more than one line for clarity. Place each directive in a web server configuration file on one line.

Local file path means a file path to the `plugin-cfg.xml` file on an spplication server that is on the same machine as the web server. Remote file path means the file path to the `plugin-cfg.xml` file when the spplication server is on a remote machine.

The Web Server Plug-ins Configuration Tool installs a dummy `plugin-cfg.xml` file during installation, but this file requires periodic propagation from the real file on the spplication server machine.

The web server definition name in the following examples is webserver1.

The node name *node_name* in the following spplication server local file paths is *web_server_name*`_node` for a standalone spplication server or *managed_node_name* for a managed node.

Procedure

- AIX Configure entries in the `httpd.conf` file.

  Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

  ```
  LoadModule
       was_ap20_module /usr/IBM/WebSphere/Plugins/bin/mod_was_ap20_http.so
  ```

  Local distributed example:

  ```
  WebSpherePluginConfig
       profile_root/config/cells/
          dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
  ```

  Example:

  ```
  WebSpherePluginConfig
       /usr/IBM/WebSphere/Plugins/config/webserver1/plugin-cfg.xml
  ```

- Linux Solaris Configure entries in the `httpd.conf` file.

  Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

  ```
  LoadModule
       was_ap20_module /opt/IBM/WebSphere/Plugins/bin/mod_was_ap20_http.so
  ```

  Local distributed example:

  ```
  WebSpherePluginConfig
       profile_root/config/cells/
          dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
  ```

  Example:

  ```
  WebSpherePluginConfig
       /opt/IBM/WebSphere/Plugins/config/webserver1/plugin-cfg.xml
  ```

- HP-UX Configure entries in the `httpd.conf` file.

  Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

  ```
  LoadModule
       was_ap20_module /opt/IBM/WebSphere/Plugins/bin/mod_was_ap20_http.sl
  ```

  Local distributed example:

  ```
  WebSpherePluginConfig
       profile_root/config/cells/
          dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
  ```

Example:

```
WebSpherePluginConfig
     /opt/IBM/WebSphere/Plugins/config/webserver1/plugin-cfg.xml
```

- Windows Configure entries in the `httpd.conf` file.

  Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

  ```
  LoadModule was_ap20_module
       drive:\IBM\WebSphere\Plugins\bin\mod_was_ap20_http.dll
  ```

  Local distributed example:

  ```
  WebSpherePluginConfig
       profile_root\config\cells\
          dmgrcell\nodes\managednode\servers\webserver1\plugin-cfg.xml
  ```

  Example:

  ```
  WebSpherePluginConfig
       C:\IBM\WebSphere\Plugins\config\webserver1\plugin-cfg.xml
  ```

Results

This procedure results in editing and re-configuring IBM HTTP Server powered by Apache 2.x.

What to do next

If the IBM HTTP Server 1.3.2x directive, LoadModule ibm_app_server_http_module, is present in an IBM HTTP Server 2.x `httpd.conf` file, the IBM HTTP Server cannot start. You must comment or delete the directive to start the Version 2.x server.

The mod_was_ap20_http plug-in module requires the GSKIT Secure Sockets Layer (SSL) encryption library if the plug-in is configured to support encrypted connections to back-end application servers.

Installing the web server plug-ins installs the GSKIT SSL encryption library at the required level if it is not installed. If you manually copy the plug-in to a new machine, you might not have the required GSKIT libraries for encrypting back-end connections.

After configuring a web server, you can install applications on it. See the Applications section of the information center for more information.

Tip: To unconfigure a web server, reverse the manual steps and remove what was manually added in this procedure.

Configuring IBM HTTP Server Version 6.x

This topic describes how to change configuration settings for IBM HTTP Server.

Before you begin

The Web Server Plug-ins Configuration Tool configures the web server plug-in.

See “Installing and configuring web server plug-ins” on page 58.

This topic describes how to configure IBM HTTP Server, Version 6.x. Other procedures in “Editing web server configuration files” on page 12 describe configuring other supported web servers.

About this task

Perform the step that configures IBM HTTP Server for your operating system.

Examples and messages are shown on more than one line for clarity. Place each directive in a web server configuration file on one line.

Local file path means a file path to the `plugin-cfg.xml` file on an spplication server that is on the same machine as the web server. Remote file path means the file path to the `plugin-cfg.xml` file when the spplication server is on a remote machine.

The Web Server Plug-ins Configuration Tool installs a dummy `plugin-cfg.xml` file during installation, but this file requires periodic propagation from the real file on the spplication server machine.

The *node_name* in the following spplication server local file paths is *web_server_name*`_node` for a standalone spplication server or *managed_node_name* for a managed node.

Procedure

- AIX Configure entries in the `httpd.conf` file.

  Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

  ```
  LoadModule
       was_ap20_module /usr/IBM/HTTPServer/Plugins/bin/mod_was_ap20_http.so
  ```

  Local distributed example:

  ```
  WebSpherePluginConfig
       profile_root/config/cells/
          dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
  ```

  Example:

  ```
  WebSpherePluginConfig
       /usr/IBM/HTTPServer/Plugins/config/webserver1/plugin-cfg.xml
  ```

- Linux Solaris Configure entries in the `httpd.conf` file.

  Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

  ```
  LoadModule
       was_ap20_module /opt/IBM/HTTPServer/Plugins/bin/mod_was_ap20_http.so
  ```

  Local distributed example:

  ```
  WebSpherePluginConfig
       profile_root/config/cells/
          dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
  ```

  Example:

  ```
  WebSpherePluginConfig
       /opt/IBM/HTTPServer/Plugins/config/webserver1/plugin-cfg.xml
  ```

- HP-UX Configure entries in the `httpd.conf` file.

  Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

  ```
  LoadModule
       was_ap20_module /opt/IBM/HTTPServer/Plugins/bin/mod_was_ap20_http.sl
  ```

  Local distributed example:

  ```
  WebSpherePluginConfig
       profile_root/config/cells/
          dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
  ```

  Example:

  ```
  WebSpherePluginConfig
       /opt/IBM/HTTPServer/Plugins/config/webserver1/plugin-cfg.xml
  ```

- Windows Configure entries in the `httpd.conf` file.

  Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

```
LoadModule was_ap20_module
     drive:\Program Files\IBM\HTTPServer\Plugins\bin\mod_was_ap20_http.dll
```

Local distributed example:

```
WebSpherePluginConfig
     profile_root\config\cells\
        dmgrcell\nodes\managednode\servers\webserver1\plugin-cfg.xml
```

Example:

```
WebSpherePluginConfig
     C:\Program Files\IBM\HTTPServer\Plugins\config\webserver1\plugin-cfg.xml
```

Results

This procedure results in editing and re-configuring IBM HTTP Server.

What to do next

If the IBM HTTP Server V1.3.x directive, LoadModule ibm_app_server_http_module, is present in an IBM HTTP Server Version 6.x and later `httpd.conf` file, IBM HTTP Server cannot start. You must comment or delete the directive to start the Version 6.x server.

The mod_was_ap20_http plug-in module requires the GSKIT Secure Sockets Layer (SSL) encryption library if the plug-in is configured to support encrypted connections to back-end WebSphere Application Servers.

Installing the web server plug-ins installs the GSKIT SSL encryption library at the required level if it is not installed. If you manually copy the plug-in to a new machine, you might not have the required GSKIT libraries for encrypting back-end connections.

After configuring a web server, you can install applications on it. See the Applications section of the information center for more information.

Tip: To unconfigure a web server, reverse the manual steps and remove what was manually added in this procedure.

Configuring IBM HTTP Server Version 7.0

This topic describes how to change configuration settings for IBM HTTP Server.

Before you begin

The Web Server Plug-ins Configuration Tool configures the web server plug-in.

This topic describes how to configure IBM HTTP Server, Version 7.0. Other procedures in “Editing web server configuration files” on page 12 describe configuring other supported web servers.

About this task

Perform the step that configures IBM HTTP Server for your operating system.

Examples and messages are shown on more than one line for clarity. Place each directive in a web server configuration file on one line.

Local file path means a file path to the `plugin-cfg.xml` file on an application server that is on the same machine as the web server. Remote file path means the file path to the `plugin-cfg.xml` file when the application server is on a remote machine.

The Web Server Plug-ins Configuration Tool installs a dummy `plugin-cfg.xml` file during installation, but this file requires periodic propagation from the real file on the application server machine.

The *node_name* in the following application server local file paths is *web_server_name*`_node` for a standalone application server, or *managed_node_name* for a managed node.

Procedure

- AIX Configure entries in the `httpd.conf` file.

  Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

  ```
  LoadModule
       was_ap22_module /usr/IBM/HTTPServer/Plugins/bin/mod_was_ap22_http.so
  ```

  Local distributed example:

  ```
  WebSpherePluginConfig
       profile_root/config/cells/
          dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
  ```

  Example:

  ```
  WebSpherePluginConfig
       /usr/IBM/HTTPServer/Plugins/config/webserver1/plugin-cfg.xml
  ```

- Linux Solaris Configure entries in the `httpd.conf` file.

  Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

  ```
  LoadModule
       was_ap22_module /opt/IBM/HTTPServer/Plugins/bin/mod_was_ap22_http.so
  ```

  Local distributed example:

  ```
  WebSpherePluginConfig
       profile_root/config/cells/
          dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
  ```

  Example:

  ```
  WebSpherePluginConfig
       /opt/IBM/HTTPServer/Plugins/config/webserver1/plugin-cfg.xml
  ```

- HP-UX Configure entries in the `httpd.conf` file.

  Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

  ```
  LoadModule
       was_ap22_module /opt/IBM/HTTPServer/Plugins/bin/mod_was_ap22_http.sl
  ```

  Local distributed example:

  ```
  WebSpherePluginConfig
       profile_root/config/cells/
          dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
  ```

  Example:

  ```
  WebSpherePluginConfig
       /opt/IBM/HTTPServer/Plugins/config/webserver1/plugin-cfg.xml
  ```

- Windows Configure entries in the `httpd.conf` file.

  Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

  ```
  LoadModule was_ap22_module
       drive:\Program Files\IBM\HTTPServer\Plugins\bin\mod_was_ap22_http.dll
  ```

  Local distributed example:

  ```
  WebSpherePluginConfig
       profile_root\config\cells\
          dmgrcell\nodes\managednode\servers\webserver1\plugin-cfg.xml
  ```

  Example:

```
WebSpherePluginConfig
     C:\Program Files\IBM\HTTPServer\Plugins\config\webserver1\plugin-cfg.xml
```

Results

This procedure results in editing and re-configuring IBM HTTP Server.

What to do next

The mod_was_ap22_http plug-in module requires the GSKIT Secure Sockets Layer (SSL) encryption library if the plug-in is configured to support encrypted connections to back-end WebSphere Application Servers.

Installing the web server plug-ins installs the GSKIT SSL encryption library at the required level if it is not installed. If you manually copy the plug-in to a new machine, you might not have the required GSKIT libraries for encrypting backend connections.

After configuring a web server, you can install applications on it. See the Applications section of the information center for more information.

Tip: To unconfigure a web server, reverse the manual steps and remove what was manually added in this procedure.

Configuring IBM HTTP Server Version 8.0

This topic describes how to change configuration settings for IBM HTTP Server.

Before you begin

The Web Server Plug-ins Configuration Tool configures the web server plug-in.

This topic describes how to configure IBM HTTP Server, Version 8.0. Other procedures in "Editing web server configuration files" on page 12 describe configuring other supported web servers.

About this task

Perform the step that configures IBM HTTP Server for your operating system.

Examples and messages are shown on more than one line for clarity. Place each directive in a web server configuration file on one line.

Local file path means a file path to the `plugin-cfg.xml` file on an application server that is on the same machine as the web server. Remote file path means the file path to the `plugin-cfg.xml` file when the application server is on a remote machine.

The Web Server Plug-ins Configuration Tool installs a dummy `plugin-cfg.xml` file during installation, but this file requires periodic propagation from the real file on the application server machine.

The *node_name* in the following application server local file paths is *web_server_name*`_node` for a standalone application server, or *managed_node_name* for a managed node.

Procedure

- AIX Configure entries in the `httpd.conf` file.

  Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

```
LoadModule
     was_ap22_module /usr/IBM/HTTPServer/Plugins/bin/mod_was_ap22_http.so
```

Local distributed example:

```
WebSpherePluginConfig
     profile_root/config/cells/
        dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
```

Example:

```
WebSpherePluginConfig
     /usr/IBM/HTTPServer/Plugins/config/webserver1/plugin-cfg.xml
```

- Linux Solaris Configure entries in the `httpd.conf` file.

  Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

```
LoadModule
     was_ap22_module /opt/IBM/HTTPServer/Plugins/bin/mod_was_ap22_http.so
```

Local distributed example:

```
WebSpherePluginConfig
     profile_root/config/cells/
        dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
```

Example:

```
WebSpherePluginConfig
     /opt/IBM/HTTPServer/Plugins/config/webserver1/plugin-cfg.xml
```

- HP-UX Configure entries in the `httpd.conf` file.

  Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

```
LoadModule
     was_ap22_module /opt/IBM/HTTPServer/Plugins/bin/mod_was_ap22_http.sl
```

Local distributed example:

```
WebSpherePluginConfig
     profile_root/config/cells/
        dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
```

Example:

```
WebSpherePluginConfig
     /opt/IBM/HTTPServer/Plugins/config/webserver1/plugin-cfg.xml
```

- Windows Configure entries in the `httpd.conf` file.

  Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

```
LoadModule was_ap22_module
     drive:\Program Files\IBM\HTTPServer\Plugins\bin\mod_was_ap22_http.dll
```

Local distributed example:

```
WebSpherePluginConfig
     profile_root\config\cells\
        dmgrcell\nodes\managednode\servers\webserver1\plugin-cfg.xml
```

Example:

```
WebSpherePluginConfig
     C:\Program Files\IBM\HTTPServer\Plugins\config\webserver1\plugin-cfg.xml
```

Results

This procedure results in editing and re-configuring IBM HTTP Server.

What to do next

The mod_was_ap22_http plug-in module requires the GSKIT Secure Sockets Layer (SSL) encryption library if the plug-in is configured to support encrypted connections to back-end WebSphere Application Servers.

Installing the web server plug-ins installs the GSKIT SSL encryption library at the required level if it is not installed. If you manually copy the plug-in to a new machine, you might not have the required GSKIT libraries for encrypting backend connections.

After configuring a web server, you can install applications on it. See the Applications section of the information center for more information.

Tip: To unconfigure a web server, reverse the manual steps and remove what was manually added in this procedure.

Configuring Microsoft Internet Information Services (IIS)

This topic describes manual configuration settings for Internet Information Services (IIS).

Before you begin

The Web Server Plug-ins Configuration Tool configures the web server. This topic describes how to configure the Internet Information Services (IIS) Web Server manually. Other procedures in "Editing web server configuration files" on page 12 describe configuring other supported web servers.

You must have read/write access to the *plugins_root* directory to perform this task.

About this task

Use the following procedure to manually reproduce how the Web Server Plug-ins Configuration Tool configures the Microsoft Internet Information Services Web Server.

Procedure

- Configure IIS Version 6.0.
  1. Start the IIS application and create a new virtual directory for the website instance that you intend to work with WebSphere Application Server. These instructions assume that you are using the Default Web Site.

     Click **Programs > Administrative Tools > Internet Information Services (IIS) Manager** on a Windows Server 2003 Standard Edition system, for example.
  2. Expand the tree on the left until you see **Default Web Site**.

     Right-click **Default Web Site > New > Virtual Directory** to create the directory with a default installation.
  3. Type **sePlugins** in the **Alias** field in the Virtual Directory Alias panel, then click **Next**.
  4. Browse to the *plugins_root*`\bin\IIS_`*web_server_name* directory in the Path field of the Web Site Content Directory panel , then click **Next**.

     For example, select the `C:\Program Files\IBM\WebSphere\Plugins\bin\IIS_webserver1` directory.
  5. Select the appropriate permission check boxes in the Virtual Directory Access Permissions panel.

     Select the **Read** check box and the **Execute (such as ISAPI applications or CGI)** check box, for example.
  6. Click **Next** to add the `sePlugins` virtual directory to your default website.
  7. Click **Finish** when the success message displays.
  8. Copy the plug-in binaries to the *plugins_root* `\bin\IIS_`*web_server_name* directory.

     For example. copy the plug-in binary files to the `C:\Program Files\IBM\WebSphere\Plugins\bin\IIS_webserver1` directory.

     The `plugin-cfg.loc` file resides in this directory. The first line of the `plugin-cfg.loc` file identifies the location of the `plugin-cfg.xml` file.
  9. Expand the **Web Sites** folder in the left pane navigation tree of the IIS Manager panel.
  10. Right-click **Default Web Site** in the navigation tree and click **Properties**.
  11. Add the Internet Services Application Programming Interface (ISAPI) filter into the IIS configuration.

      In the Default Web Site Properties panel, perform the following steps:

   a. Click the **ISAPI Filters** tab.
   b. Click **Add** to open the **Add/Edit Filter Properties** dialog window.
   c. Type **iisWASPlugin** in the **Filter name** field.
   d. Click **Browse** to select the `C:\Program Files\IBM\WebSphere\Plugins\bin\IIS_webserver1\iisWASPlugin_http.dll` file for the value of the **Executable** field.

      Browse to your *plugins_root* `\bin\IIS`*web_server_name* directory to select the **iisWASPlugin_http.dll** file.
   e. Click **OK** to close the **Add/Edit Filter Properties** dialog window.
   f. Click **OK** to close the **Default Web Site Properties** window.
12. Set the value in the `plugin-cfg.loc` file to the location of the configuration file.

    Set the location to the *plugins_root* `\config\` *`webserver_name`* `\plugin-cfg.xml` file, which might be `C:\Program Files\IBM\WebSphere\Plugins\config\IIS_webserver1\plugin-cfg.xml` file.

    The location varies depending on how you have configured your system. If the web server and the application server are on separate machines, you have a remote installation.

    If the two servers are on the same machine, you have a local installation.

    If the two servers are on the same machine and the application server is federated, you have a local distributed installation.

    Local distributed example:

    ```
    "C:\IBM\WebSphere\AppServer\profiles\custom01\config\cells\
          dmgrcell\nodes\managed_node\servers\webserver1\plugin-cfg.xml"
    ```

    Example:

    ```
    "C:\IBM\WebSphere\Plugins\config\webserver1\plugin-cfg.xml"
    ```
13. Configure the web server to run WebSphere Application Server extensions:
    a. Expand the left pane navigation tree and click on the **Web Service Extensions** folder in the IIS Manager panel.
    b. Click **Add a new web service extension** to open the **New Web Service Extension** dialog window.
    c. In the **Extension name** field, type `WASPlugin` as the name of the new web service extension.
    d. Click **Add** to open the **Add file** dialog window.
    e. In the **Path to file** field, type the path or click **Browse** to navigate to the correct `iisWASPlugin_http.dll` file that the new web service extension requires, and click **OK**.
    f. Select the **Set extension status to Allowed** check box to automatically set the status of the new web service extension to Allowed and click **OK**.

- Configure IIS Version 7.x.

Note: 2008 You can use IIS Version 7 with Windows Server 2008 and IIS Version 7.5 with Windows Server 2008 R2.

1. Install IIS Version 7.x with the necessary IIS Version 6.0 Management Compatibility components.

   IIS Version 6.0 Management Compatibility components are not automatically installed by default. Complete the following steps to install IIS Version 7.x with the necessary IIS Version 6.0 Management Compatibility components.
   – Complete the following steps to bring up the Server Manager window on Windows Server 2008:
     a. Click **Start > Administrative Tools > Server Managers**.
     b. Click **Action > Add Roles**, and then click **Next**.
     c. On the Select Server Roles page, select the Web Server (IIS) role, and then click **Next.**
     d. If a prompt for the Windows Process Activation Service feature displays, click **Add Feature > Next**, and then click **Next** on the IIS introduction page
   – When the Role Services window displays, verify that the following options are selected in addition to the default options that are already selected.

- Internet Information Services: Management Tools
- IIS Version 6.0 Management Compatibility: IIS Version 6.0 Management Console, IIS Version 6.0 Scripting Tools, IIS Version 6.0 WMI Compatibility, and IIS Metabase compatibility
- Application Development: ISAPI Extensions, ISAPI Filters

– Click **Next** to enable the selected options, and then click **Install** on the next window to perform the installation.

– When the installation finishes, click **Close** on the Installation Results window.

2. Install the web server plug-ins.

   If you are using an already installed web server plug-in, go to the next step, and re-configure IIS Version 7.x to use that web server plug-in.

3. Optional: Re-configure IIS Version 7.x if the web server plug-in is already installed:

   The following steps are completed automatically during web server plug-in installation. You only need to complete these steps are if you are re-configuring IIS Version 7.x to use an existing web server plug-in.

   Complete the following steps to re-configure IIS Version 7.x:

   a. Click **Start > All Programs > Administrative Tools > Internet Information Services (IIS) Manager** on a Windows Server 2008 operating system. This action starts the IIS application, and creates a new virtual directory for the website instance that you intend to use with WebSphere Application Server. These instructions assume that you are using the default website.
   b. Expand the tree on the left until you see Default Web Site.
   c. Right-click **Default Web Site > Add Virtual Directory** to create the directory with a default installation.
   d. Type sePlugins in the **Alias** field on the Virtual Directory Alias window.
   e. Browse to the *plugins_root*`\bin\IIS_`*web_server_name* directory in the **Physical Path** field of the Web Site Content Directory window, and then click **OK**. For example, select the `C:\Program Files\IBM\WebSphere\Plugins\bin\IIS_webserver1` directory.
   f. Click the Test Settings button. If the settings test fails, then either change the permissions of the physical directory, or select **Connect As**, and let IIS connect as a Windows user account that has authority to files in that physical path.

      Attention: When you click the Test Settings button, you might encounter the following warning message if you use the default "Pass-thru authentication" setting:

      ```
      Cannot verify access to path
      ```

      For more information, refer to the Microsoft information on this subject.
   g. Click **OK** to add the sePlugins virtual directory to your default website.
   h. In the navigation tree, select the sePlugins virtual directory that you just created.
   i. On the Features panel, double-click **Handler Mappings**, and then click **Edit Feature Permissions** on the Actions panel.
   j. Select **Script** and **Execute**, if they are not already selected.
   k. Click **OK**.
   l. Manually copy the plug-in binaries to the *plugins_root*`\bin\IIS_`*web_server_name* directory. For example, copy the plug-in binary files to the `C:\Program Files\IBM\WebSphere\Plugins\bin\IIS_webserver1` directory.

      The plugin-cfg.loc file resides in this directory. The first line of the plugin-cfg.loc file identifies the location of the plugin-cfg.xml file.
   m. Return to the IIS Manager window, and expand the Web Sites folder in the left-hand navigation tree of that window.
   n. Select **Default Web Site** in the navigation tree.

o. Add the Internet Services Application Programming Interface (ISAPI) filter into the IIS configuration.

   On the Default Web Site Properties panel, complete the following steps:

   1) Double-click the ISAPI Filters tab.
   2) Click to open the Add/Edit Filter Properties dialog window.
   3) Type iisWASPlugin in the Filter name field.
   4) Click **Browse** to select the plug-in file located in the *plugins_root* `\bin\IIS_`*web_server_name*\iisWASPlugin_http.dll directory.
   5) Click **OK** to close the Add/Edit Filter Properties dialog window.

p. In the navigation tree, select the top level server node.

q. On the Features panel, double-click **ISAPI and CGI Restrictions**, and then, on the Actions panel, click **Add**.

   To determine the value to specify for the **ISAPI or CGI Path** property, browse to, and then select the same plug-in file that you selected in the previous step. For example:

   ```
   plugins_root\bin\IIS_web_server_name\iisWASPlugin_http.dll
   ```

   Then type WASPlugin in the **Description** field, select **Allow extension path to execute**, and then click **OK** to close the **ISAPI and CGI Restrictions** dialog window.

r. Set the value in the plugin-cfg.loc file to the location of the configuration file at *plugins_root* `\config\`*webserver_name*`\plugin-cfg.xml`.

   Following is the default location:

   ```
   C:\Program Files\IBM\WebSphere\Plugins\config\IIS_webserver1\plugin-cfg.xml
   ```

   The location varies depending on how you have configured your system. If the web server, and WebSphere Application Server are on separate machines, you have a remote installation. If the web server, and WebSphere Application Server are on the same machine, then you have a local installation, and the correct location of the configuration file might be set. If the two servers are on the same machine, and the application server is federated, you have a local distributed installation.

   Local distributed example:

   ```
   C:\IBM\WebSphere\AppServer\profiles\custom01\config\cells\dmgrcell\nodes
      \managed_node\servers\webserver1\plugin-cfg.xml
   ```

   Local example:

   ```
   C:\IBM\WebSphere\Plugins\config\webserver1\
   plugin-cfg.xml
   ```

s. Restart IIS Version 7.x and your WebSphere Application Server profile.

- Enable IIS Version 6.0 or IIS Version 7.x to communicate with a web server plug-in that is running in 32–bit mode.

  The web server plug-in for IIS is available in both 32-bit, and 64-bit versions. When using the 32-bit version plug-in on a Microsoft Windows 64-bit operating system, the following steps should be taken to enable the native 64-bit IIS to run the plug-in under a 32-bit worker process.

  The Windows Server TechNet topic Running 32-bit Applications on 64-bit Windows describes how to enable the native 64-bit IIS Version 6.0 to run the web server plug-in under a 32-bit worker process.

  Complete the following steps to enable the native 64-bit IIS Version 7.x to run the web server plug-in under a 32-bit worker process:

  1. Launch the IIS Version 7.x administrative console.
  2. On the connections page, expand the **Sites** node, and select the website that is intended for the web server plug-in.
  3. On the actions page, click **Basic Settings**, and make a note of the Application Pool name.
  4. Click **Cancel** , and then select the **Application Pools** node on the connections page.

5. On the features page, right-click the application pool that you noted in the earlier step, and then choose **Advanced Settings**.
6. Set the **Enable 32-bit Applications** property to `True`.
7. Click **OK** to complete the configuration change.
8. Restart the corresponding application pool.

- Optional: Configure multiple websites. Given:
  - There are two websites defined: website1, website2.
  - The DLL files are already created as `bin/website1/iisWASPlugin_http.dll` and `bin/website2/iisWebsite2/iisWASPlugin_http.dll`.
  - The `plugin-cfg.loc` files are created in the same folder as the DLL files.

1. Run IIS in worker process isolation mode (default).

   To enable worker process in isolation mode:

   a. Open the IIS Manager console and expand the local computer by clicking the **plus sign**.
   b. Expand the **Web Sites** folder, then right-click the **Default Web Sites** folder.
   c. Click **Properties**, then click the **Service** tab.
   d. Under Isolation mode, clear the Run web service in IIS 5.0 isolation mode check box to enable worker process isolation mode.
2. Define two application pools; one for website1 and the other for website2. Do not use the pre-defined application pool DefaultAppPool.
3. Define the two websites, including the filter setting, virtual host setting, and extension settings.
4. Assign an application pool for each website.
   a. Under each website folder, right click on the ***website name***.
   b. Click **Property**, and select the **Home Directory** tab. 2.
   c. At the bottom of the application settings, select the application pool you defined for website 1 from the drop-down list of application pools.
   d. Click **OK**.
   e. Repeat the previous steps for the second website and select the application pool you defined for website 2.
5. Start the IIS service and start each website.

Results

This procedure results in re-configuring the Internet Information Services (IIS) Web Server.

Note: On some editions of the Windows operating system, the `http_plugin.log` file is not created automatically when the plug-in is installed and the IIS Web Server is started. If the `http_plugin.log` file is not created after performing the procedure described above, take the following steps:
1. Open a Windows Explorer window.
2. Browse to the *plugins_root*`\logs\`*`web_server_name`* directory.
3. Share the folder and give full-control permission to everyone.

What to do next

You can now install applications on the configured Web server. See the Applications section of the information center for more information.

Tip: To unconfigure a web server, reverse the manual steps and remove what was manually added in this procedure.

Configuring the Sun Java System Web Server

This topic describes how to change configuration settings for the Sun Java System Web Server (formerly Sun ONE Web Server and iPlanet Web Server), Version 6.0 and later.

Before you begin

The Web Server Plug-ins Configuration Tool configures the web server plug-in. This topic describes how to configure the Sun Java System Web Server if you must change something in the existing configuration. Other procedures in “Editing web server configuration files” on page 12 describe configuring other supported web servers.

About this task

Configure the Sun ONE Web Server 6.0 or Sun Java System Web Server, Version 6.1 and later.

Examples and messages are sometimes shown on more than one line for ease of presentation. Verify that each directive in a web server configuration file is on one line.

Procedure

1. Configure entries in the `obj.conf` configuration file and in the `magnus.conf` configuration file for Version 6.0 and later of Sun Java System Web Server.
   a. Add two directives to the `obj.conf` file after the `<Object name=default>` tag:

```
Service fn="as_handler"
AddLog fn="as_term"
```

   b. Add two directives at the end of the `magnus.conf` file:

      The location for the `bootstrap.properties` directive varies, depending on how you have configured your system. If the Web server and the application server are on separate machines, you have a remote installation.

      If the two servers are on the same machine, you have a local installation.

      If the two servers are on the same machine and the application server is a managed node, you have a local distributed installation.

      gotcha: The following examples reference the libns61_http.so file, which is only supported for the Sun Java System Web Server 7.0 and 6.1. If you are using a Sun ONE Web Server 6.0, change these references to libns41_http.so.

      - AIX HP-UX Linux Solaris

        Local distributed example:

```
Init fn="load-modules"
     funcs="as_init,as_handler,as_term"
     shlib="/opt/IBM/WebSphere/Plugins/bin/libns61_http.so"
Init fn="as_init"
     bootstrap.properties="profile_root/config/
        cells/dmgrcell/nodes/managed_node/servers/webserver1/plugin-cfg.xml"
```

        Example:

```
Init fn="load-modules"
     funcs="as_init,as_handler,as_term"
     shlib="/opt/IBM/WebSphere/Plugins/bin/libns61_http.so"
Init fn="as_init"
     bootstrap.properties="/opt/IBM/WebSphere/Plugins/config/webserver1/plugin-cfg.xml"
```

      - Windows

        Local distributed example:

```
Init fn="load-modules"
     funcs="as_init,as_handler,as_term"
     shlib="C:\IBM\WebSphere\Plugins\bin\ns41_http.dll"
```

```
Init fn="as_init"
     bootstrap.properties=
     "profile_root\config\cells\
        dmgrcell\nodes\managed_node\servers\webserver1\plugin-cfg.xml"
```

Example:

```
Init fn="load-modules"
     funcs="as_init,as_handler,as_term"
     shlib="C:\IBM\WebSphere\Plugins\bin\ns41_http.dll"
Init fn="as_init"
     bootstrap.properties="C:\IBM\WebSphere\Plugins\config\webserver1\plugin-cfg.xml"
```

2. Set the shared library path on HP-UX machines. On some installations of Sun Java System Web Server on an HP-UX machine, it is necessary to manually set the `SHLIB_PATH` variable to `/usr/lib` before starting Sun Java System Web Server with a plug-in that is configured for Secured Sockets Layer (SSL). For example, in the korn shell, issue the following command before invoking the command to start the Sun Java System Web Server:

   ```
   export SHLIB_PATH=/usr/lib:$SHLIB_PATH
   ```

3. Disable the feature of Sun Java System Web Server Version 6.1 that supports servlets and JavaServer Pages files by default. Disable this feature so that the WebSphere Application Server plug-in can handle the requests.

   Perform the following steps to disable the feature:

   a. Remove or comment out the following two lines from the `obj.conf` configuration file:

      ```
      NameTrans fn="ntrans-j2ee" name="j2ee"
      Error fn="error-j2ee"
      ```

   b. Remove or comment out the following line from the `magnus.conf` configuration file: AIX HP-UX Linux Solaris

      ```
      Init fn="load-modules"
          shlib="C:/Sun/WebServer6.1/bin/https/bin/j2eeplugin.so"
             shlib_flags="(global|now)"
      ```

      Windows

      ```
      Init fn="load-modules"
          shlib="C:\Sun\WebServer6.1\bin\https\bin\j2eeplugin.dll"
             shlib_flags="(global|now)"
      ```

4. If you are configuring a 64-bit version of the Sun Java System Web Server, modify the loadModule entry in the `magnus.conf` configuration file to point to the `*plugin_root*/bin/64bit/libns61_http.so` file.

Results

This procedure results in editing and re-configuring the Sun Java System Web Server.

What to do next

After configuring a web server, you can install applications on it. See the Applications section of the information center for more information.

Tip: To unconfigure a web server, reverse the manual steps and remove what was manually added in this procedure.

Creating web server templates

A web server template is used to define the configuration settings for a new web server. When you create a new web server, you can either create a new web server definition or use a previously created web server template that is based on another, already existing web server.

About this task

To create a web server template.

Procedure

1. In the administrative console, click **Servers > Server Types > Web servers >** , and then click **Templates**.

   You can also use the **createWebServerTemplate** command for the AdminTask object to create a web server template.
2. On the Server Templates page, click **New**.
3. Select the web server that you want to use to create the new template, and then click **OK**.
4. Enter the name of the new template and, optionally, a description of that template that distinguishes it from your other templates.
5. Click **OK**.

Results

Your new template displays in the list of server templates that you can use to create a new web server.

What to do next

You can perform one of the following actions to display a list of all of the server templates that are available on your system:

- In the administrative console, click **Servers > Server Types > Web servers**, and then click **Templates**.
- Issue a **listServerTemplates** wsadmin command that includes the `-serverType WEB_SERVER` parameter.

Allowing web servers to access the administrative console

This topic describes how to add the virtual host that servers the administrative console to the plug-in configuration file so that you can access the administrative console through a web server.

Before you begin

Note: AIX HP-UX Linux Solaris If a web server is in a configuration, its port must be higher than 1023 to use a nonroot node agent. Otherwise the node agent must be running as root in order for the administrative console of the deployment manager to stop and start the web server process in that managed node.

Install your WebSphere Application Server product, a web server, the Web Server Plug-ins, and the WebSphere Customization Toolbox.

The Web Server Plug-ins Configuration Tool creates a web server definition on the application server system, either directly when they are on the same machine or by a script for remote scenarios.

After creating the web server definition, the plug-in configuration file exists within the web server definition.

The `plugin-cfg.xml` file can be overwritten by the deployment manager synchronization operation, the GenPluginCfg script or any other method that regenerates the file. If you make changes to the `plugin-cfg.xml` file, and want to keep those changes, it is recommended that you create a copy of the file in a separate location. Make your manual updates each time the file is automatically refreshed by another process.

About this task

This task gives you the option of configuring the admin_host so that web servers can access the administrative console. When the web server plug-in configuration file is generated, it does not include admin_host on the list of virtual hosts.

Procedure

1. Use the administrative console to change the admin_host virtual host group to include the web server port (80 by default).
   a. Click **Environment > Virtual Host > admin_host > Host Aliases > New**.

      The default port that displays is 80, unless you specify a different port during profile creation.
   b. Specify the IP address, or the name of the machine that is hosting the HTTP server.

      For example, if you installed a WebSphere Application Server product on a machine that is named waslwaj.rtp.ibm.com, specify the name in this field.
2. Click **Apply > Save**.
3. Stop and restart the application server.

   For example, to access the administrative console of a stand-alone application server, stop and restart the server1 process.

   To stop server1, open a command window and navigate to the *profile_root*/`bin` directory. Then issue the following command:
   ```
   ./stopServer.sh server1
   ```
   After receiving the following message, you can restart the application server:
   ```
   Server server1 stop completed.
   ```
   To start the application server, issue the following command:
   ```
   ./startServer.sh server1
   ```
   When you receive a message that is similar to the following message, the server1 process is running:
   ```
   Server server1 open for e-business; process id is 1719
   ```
4. Stop and restart a deployment manager.

   For example, to access the administrative console of a deployment manager, stop and restart the deployment manager.

   To stop the deployment manager, open a command window and navigate to the *profile_root*/`bin` directory. Then issue this command:
   ```
   ./stopManager.sh
   ```
   Then issue the following command to stop the deployment manager:
   ```
   ./stopManager.sh
   ```
   After receiving the following message, you can restart the deployment manager:
   ```
   Server dmgr stop completed.
   ```
   To start the deployment manager, issue the following command:
   ```
   ./startManager.sh
   ```
   When you receive a message that is similar to the following message, the deployment manager is running:
   ```
   Server dmgr open for e-business; process id is 1720
   ```
5. Edit the `plugin-cfg.xml` file to include the following entries:
   ```
   <VirtualHostGroup Name="admin_host">
           <VirtualHost Name="*:9060"/>
           <VirtualHost Name="*:80"/>
           <VirtualHost Name="*:9043"/>
       </VirtualHostGroup>
       ...
       ...
   ```

```
...
<ServerCluster Name="server1_SERVER1HOSTserver1_Cluster">
    <Server LoadBalanceWeight="1" Name="SERVER1HOSTserver1_dmgr">
        <Transport Hostname="SERVER1HOST" Port="9060" Protocol="http"/>
    </Server>

    <PrimaryServers>
        <Server Name="SERVER1HOSTserver1_dmgr"/>
    </PrimaryServers>
</ServerCluster>
...
...
...
<UriGroup Name="admin_host_server1_SERVER1HOSTserver1_Cluster_URIs">
    <Uri AffinityCookie="JSESSIONID"
        AffinityURLIdentifier="jsessionid" Name="/ibm/console/*"/>
</UriGroup>
<Route ServerCluster="server1_SERVER1HOSTserver1_Cluster"
    UriGroup="admin_host_server1_SERVER1HOSTserver1_Cluster_URIs" VirtualHostGroup="admin_host"/>
```

If your HTTP server has an HTTP port other than 80, add an entry to the VirtualHostGroup:

```
<VirtualHost Name="*:port"/>
```

The *port* variable is your HTTP server port.

Results

You can configure your supported web servers to access the administrative console application of a deployment manager or a stand-alone application server.

Administering web servers from the administrative console

Web server definition

To administer or manage a web server using the administrative console, you must create a web server definition or object in the WebSphere Application Server repository.

The creation of this object is exclusive of the actual installation of a web server. The web server object in the WebSphere Application Server repository represents the web server for administering and managing the web server from the administrative console.

The web server object contains the following web server properties:

- installation root
- port
- configuration file paths
- log file paths

In addition to web server properties, the web server contains a plug-in object. The plug-in object contains properties that define the `plugin-cfg.xml` file.

The definitions of the web server object are made using the **wsadmin** command or the administrative console. You can also define a web server object in the WebSphere Application Server repository using the profile create script during installation, a `.jacl` script, and by using the administrative console wizard.

There are three types of WebSphere Application Server nodes upon which you can create a web server. The type depends on the version of WebSphere Application Server, as follows:

- **Managed node**. A node that contains a node agent. This node can exist only in a deployment manager environment. The importance of defining a web server on a managed node is that the administration and configuration of the web server is handled through the node agent from the administrative console. Support for administration and configuration through the administrative console is limited to IBM HTTP

Server only. Non-IBM HTTP Server web servers must be on a managed node to handle plug-in administrative functions and the generation and propagation of the `plugin-cfg.xml` file.

- **Stand-alone node**. A node that does not contain a node agent. This node usually exists in WebSphere Application Server (base) or WebSphere Application Server, Express environment. A stand-alone node can become a managed node in a deployment manager environment after the node is federated . A stand-alone node does not contain a node agent, so to administer and manage IBM HTTP Server, there must be an IBM HTTP Server administration server installed and running on the stand-alone machine that the node represents. IBM HTTP Server ships with the IBM HTTP Server administration server and is installed by default. Support for administration and configuration through the administrative console is limited to IBM HTTP Server only.
- **Unmanaged node**. A node that is not associated with a WebSphere Application Server node agent. This node cannot be federated. Typically, the unmanaged node represents a remote machine that does not have WebSphere Application Server installed. However, you can define an unmanaged node on a machine where WebSphere Application Server is installed. This node can exist in aWebSphere Application Server (base), WebSphere Application Server, Express, or deployment manager environment. An unmanaged node does not contain a node agent, so to administer and manage IBM HTTP Server, an IBM HTTP Server administration server must be installed and running on the stand-alone machine that the node represents. Support for administration and configuration through the administrative console is limited to IBM HTTP Server only.

Web servers, which are not IBM HTTP Servers for WebSphere Application Server, are not fully administered from the WebSphere Application Server administrative console. The administration functions for Web servers, which are not IBM HTTP Servers for WebSphere Application Server, are:

- On managed nodes:
  - Web server status in the web server collection panel or `serverStatus.sh`
  - Generation of the `plugin-cfg.xml`
  - Propagation of the `plugin-cfg.xml`
- On unmanaged nodes:
  - Web server status in the web server collection panel or `serverStatus.sh`
  - Generation of the `plugin-cfg.xml`

Web server configuration

Plug-in configuration involves configuring the web server to use the binary plug-in module that WebSphere Application Server provides. Plug-in configuration also includes updating the plug-in XML configuration file to reflect the current application server configuration. The binary module uses the XML file to help route web client requests.

After installing a supported web server, you must install a binary plug-in module for the web server by installing the Web Server Plug-ins. The plug-in module lets the web server communicate with the application server. The Web Server Plug-ins Configuration Tool allows you to configure the web server and create a web server definition in the configuration of the application server. The Web Server Plug-ins Configuration Tool uses the following files to configure a plug-in for the web server that you select:

- The **web server configuration file** on the web server machine, such as the `httpd.conf` file for IBM HTTP Server.
- The **binary web server plug-in file** on the web server machine.
- The **plug-in configuration file, `plugin-cfg.xml`**, on the application server machine that you propagate (copy) to a Web server machine.
- The **default (temporary) plug-in configuration file, `plugin-cfg.xml`**, on the web server machine.
- The **`configure`*`web_server_name`* script** that you copy from the web server machine to the application server machine.

See the following descriptions of each file.

Web server configuration file

The web server configuration file is installed as part of the web server.

The Web Server Plug-ins Configuration Tool must re-configure the configuration file for a supported web server.

Configuration consists of adding directives that identify file locations of two files:
- Binary web server plug-in file
- Plug-in configuration file, `plugin-cfg.xml`

Binary web server plug-in file

An example of a binary plug-in module is the `mod_was_ap22_http.dll` file for IBM HTTP Server on the Windows platform.

The binary plug-in file does not change. However, the configuration file for the binary plug-in is an XML file. The application server changes the configuration file when certain changes to your WebSphere Application Server configuration occur.

The binary module reads the XML file to adjust settings and to route requests to the application server.

Plug-in configuration file, `plugin-cfg.xml`

The plug-in configuration file is an XML file with settings that you can tune in the administrative console. The file lists all of the applications installed on the web server definition. The binary module reads the XML file to adjust settings and to route requests to the application server.

The standalone application server regenerates the `plugin-cfg.xml` file in the *profile_root*/`config/cells/`*cell_name*/`nodes/`*web_server_name*`_node/servers/`*web_server_name* directory. Regeneration occurs whenever a change occurs in the application server configuration that affects deployed applications.

The deployment manager regenerates the `plugin-cfg.xml` file in the *profile_root*/`config/cells/`*cell_name*/`nodes/`*node_name*/`servers/`*web_server_name* directory whenever a change occurs in application server configuration that affects deployed applications on the managed node.

After regeneration, propagate (copy) the file to the web server machine. The binary plug-in then has access to the most current copy of its configuration file.

The web server plug-in configuration service automatically regenerates the `plugin-cfg.xml` file after certain events that change the configuration. The configuration service automatically propagates the `plugin-cfg.xml` file to an IBM HTTP Server machine when the file is regenerated. You must manually copy the file on other web servers.

Default plug-in configuration file, `plugin-cfg.xml`

The Web Server Plug-ins Configuration Tool creates the temporary `plugin-cfg.xml` file in the *plugins_root*/`config/`*web_server_name* directory. The tool creates the file for every remote installation scenario.

The default file is a placeholder that you must replace with the `plugin-cfg.xml` file from the web server definition on the application server. The default file is a replica of the file that the application server creates for a default standalone application server.

Run the `configure`*`web_server_name`* script from the *app_server_root*/`bin` directory of the application server machine for a remote installation or directly from the *plugins_root*/`bin` directory for a local installation. The script creates the web server definition in the configuration files of the default profile. To configure a different profile than the default, edit the `configure`*`web_server_name`* script. Use the -profileName parameter to identify a profile other than the default profile.

After the web server definition is created, the web server plug-in configuration service within the application server creates the first `plugin-cfg.xml` file in the web server definition on the application server machine. If you install an application, create a virtual host, or do anything that changes the configuration, you must propagate the updated `plugin-cfg.xml` file from the application server machine to the web server machine to replace the default file.

Configure*web_server_name* script for the web server definition

The Web Server Plug-ins Configuration Tool creates the `configure`*`web_server_name`* script on the web server machine in the *plugins_root*/`bin` directory. If one machine in a remote scenario is running under an operating system like AIX or Linux and the other machine is running under Windows, use the script created in the *plugins_root*/`bin/crossPlatformScripts` directory. The script is created for remote installation scenarios only.

Copy the script from the web server machine to the *app_server_root*/`bin` directory on a remote application server machine. You do not have to copy the script on a local installation. Run the script to create a web server definition in the configuration of the application server.

When using the IBM HTTP Server, configure the IBM HTTP Administration Server also. The IBM HTTP Administration Server works with the administrative console to manage web server definitions. Also, use the administrative console to update your web server definition with remote web server management options. Click **Servers > Server Types > Web servers > *web_server_name*** to see configuration options. For example, click **Remote Web server management** to change such properties as:
- Host name
- Administrative port
- User ID
- Password

Important: Always open a new command window before running this script. You can avoid a potential problem by doing so.

The problem is a potential conflict between a shell environment variable, the WAS_USER_SCRIPT environment variable, and the actual default profile. The script always works against the default profile. If the WAS_USER_SCRIPT environment variable is set, however, a conflict arises as the script attempts to work on the profile identified by the variable.

The variable is easy to set accidentally. Issue any command from the *profile_root*/`bin` directory of any profile and the variable is set to that profile.

If you have more than one profile on your system, the potential exists that the default profile and the profile identified by the variable are different profiles. If so, a conflict occurs and the script might not create the web server definition in the correct profile, or might not create the web server definition at all.

Reset the variable in either of two ways:
- Close the command window where the variable is set and open a new one.

- Change directories to the *profile_root*/`bin` directory of the default profile and source the `setupCmdLine.sh` script:

  Windows
  1. Open a command prompt window.
  2. Change directories to the *app_server_root*\`bin` directory.
  3. Issue the `setupCmdLine.bat` command.

  AIX HP-UX Linux Solaris
  1. Open a command shell window.
  2. Change directories to the *app_server_root*/`bin` directory.
  3. Issue the `. ./setupCmdLine.sh` command. Notice the space between the periods. The special format for this command sources the command to make the setting active for all processes started from the command shell.

If a web server definition already exists for a standalone application server, running the script does not add a new web server definition. Each standalone application server can have only one web server definition.

You cannot use the administrative console of a standalone application server to add or delete a Web server definition. However, you can do both tasks using the administrative scripting interface:
- Add a web server definition through the wsadmin facility using the configure*web_server_name* script. The script uses a Java Command Language (Jacl) script named `configureWebserverDefintion.jacl` to create and configure the web server definition.
- Delete a web server definition using wsadmin commands. The Web server is named webserver1 in the following example:

```
set webserverName webserver1
set webserverNodeSuffix _node
set webserverNodeName   $webserverName$webserverNodeSuffix
$AdminConfig remove [$AdminConfig getid /Node:$webserverNodeName/Server:$webserverName]
$AdminConfig remove [$AdminConfig getid /Node:$webserverNodeName]
$AdminConfig save
```

A managed node, on the other hand, can have multiple web server definitions. The script creates a new Web server definition unless the web server name is the same.

Replacing the default plug-in configuration file with the file from the web server definition (propagation)

The default file uses fixed parameter values that might not match the parameter values in the actual file on the application server. The default file is a placeholder only.

The file cannot reflect changes that occur in the application server configuration. The file also cannot reflect nondefault values that might be in effect on the application server.

The application server must have the following values in the actual `plugin-cfg.xml` file. If so, the default file can successfully configure the binary plug-in module. Then, the plug-in module can successfully communicate with the web server and the application server.

Suppose that the application server does not have the following values in the actual `plugin-cfg.xml` file. In that case, the default file configures the binary plug-in module incorrectly. The plug-in module can always communicate with the web server. But with an improper configuration file, the plug-in module cannot communicate successfully with the application server.

The following are fixed parameter values in the temporary plug-in configuration file.
- **Virtual host name**

  Default value: default_host

This virtual host is configured to serve the DefaultApplication. This value is probably the same as the value in the real `plugin-cfg.xml` file. However, suppose that you create another virtual host for serving applications and install the DefaultApplication on it. If so, the actual `plugin-cfg.xml` file is regenerated. The web server cannot access the DefaultApplication. (The application includes the snoop servlet and the hitcount servlet.)

To access applications on the new virtual host, propagate the real `plugin-cfg.xml` file. Propagation is copying the updated file from the application server machine to the web server machine.

- **HTTP transport port**

  Default value: 9080

  The `9080` value is the default value for the HTTP transport port for the default_host virtual host. This value is probably the same as the value in the updated file. However, this value changes for every profile on the application server machine. The HTTP transport port value must be unique for every application server.

  To communicate over a different port, propagate the real `plugin-cfg.xml` file.
- **Web server listening port**

  Default value: 80

  The `80` value is the default value for the port that controls communication with the web server. However, each application server profile must have a unique port value to communicate to a web server. The actual port value might be 81 or another number.

  To communicate over a different port, propagate the real `plugin-cfg.xml` file.
- **HTTPS transport port**

  Default value: 9443

  The `9443` value is the default value for the HTTPS (secure) transport port for the default_host virtual host. This value is probably the same as the value in the updated file. However, this value changes for every profile on the application server machine. The HTTPS transport port value must be unique for every application server.

  To communicate over a different secure port, propagate the real `plugin-cfg.xml` file.
- **Applications installed on the server1 application server**

  All of the default servlets and applications are included in the default file.

  To serve an application that you developed with the web server, propagate the real `plugin-cfg.xml` file.

Web server collection

Use this page to configure, manage, and view information about your web servers.

Web servers

To view this administrative console page click **Servers > Server Types > Web Servers**.

To create a new web server, click **New** to launch the Create new web server entry wizard. To manage an installed web server, select the check box beside the application name in the list and click a button:

Button	Resulting Action
Generate Plug-in	When the plug-in configuration service is enabled, a plug-in configuration file is automatically generated for a web server whenever: • The WebSphere Application Server administrator defines new web server. • An application is deployed to an Application Server. • An application is uninstalled. • A virtual host definition is updated and saved.

Button	Resulting Action
Propagate Plug-in	Choosing this action will copy the `plugin-cfg.xml` file from the local directory where the Application Server is installed to the remote machine. If you are using IBM HTTP Server V6 or higher for your web server, WebSphere Application Server can automatically propagate the plug-in configuration file to remote machines provided there is a working HTTP transport mechanism to propagate the file.
New	Launches the wizard to create a new web server entry.
Delete	Deletes one or more of the selected web server entries.
Templates ...	Opens the web server templates list panel. From this panel you can create a new template or delete existing templates.
Start	Starts one or more of the selected web servers.
Stop	Stops one or more of the selected web servers.
Terminate	Terminates one or more of the selected web servers.

Name

Specifies a logical name for the web server. This can be the host name of the machine, or any name you choose.

Web server type

Indicates the type of web server you are using.

Node

Specifies a logical name for the web server. This can be the host name of the machine, or any name you choose.

Specifies the name of the node on which the web server is defined. This column only applies for the WebSphere Application Server, Network Deployment product.

Version

Specifies the version of the WebSphere Application Server on which the web server is defined.

Status

Indicates whether the web server is started, stopped, or unavailable.

If IBM HTTP Server is defined on an unmanaged node, you must start the IBM HTTP Server administration server before you can start and stop IBM HTTP Server.

If the status is unavailable, the node agent or IBM HTTP Server administration server is not running in that node, and you must start the node agent before you can start the web server.

Table 3. Status indicators. The following table describes the status indicators.

➡	**Started**	The web server is running.
✖	**Stopped**	The web server is not running.
?	**Unknown**	Status cannot be determined. A web server with an unknown status might, in fact, be running but has an unknown status because the application server that is running the administrative console cannot communicate with this web server.

Web server configuration

Use this page to configure web server properties.

Web servers:

To view this administrative console page click **Servers > Server Types > Web Servers >** *web_server_name*.

Web server name	Specifies a logical name for the web server.
Type	Specifies the vendor of the web server. The default value is IBM HTTP Server. The options for the type of web servers are: • IHS • APACHE • IIS • SUNJAVASYSTEM • DOMINO
Port	The port from which to ping the status of the web server. This field is required. You can use the WebSphere Application Server administrative console to check if the web server is started by sending a ping to attempt to connect to the web server port that is defined. In most cases the port is 80. If you have a firewall between the web servers and application servers, you will not use port 80 on the firewall between the two systems. In most cases, your port will be different, such as 9080 or 9443. You should set the alternate ports for the web server using the WebSphere Application Server administrative console, then set that port to the web server to listen on, in addition to the typical port 80 and 443.
Installation path	Enter the fully qualified path where the web server is installed. This field is required if you are using IBM HTTP Server. For all other web servers, this field is not required. If you enable any administrative function for non-IBM HTTP Server Web servers, the installation path will be necessary.
Configuration file name	There are two ways to view or modify the contents of the configuration file: 1. Click **Edit** to view the configuration file. You will be able to make modifications from this view. This is valid for IBM HTTP Server only. 2. Click **Configuration file** under Additional properties. You will be able to make modifications from this view. This is valid for IBM HTTP Server only.
Service name - Microsoft Windows operating systems only	Specifies the Microsoft Windows operating system name for the Web server. The name is the service name and you can find it by opening the **General** properties tab of the web server service name.

Checking your IBM HTTP Server version

At times, you might need to determine the version of your IBM HTTP Server installation.

About this task

The following information describes how you can determine the IBM HTTP Server version and provides examples. The server versions that are provided in the output below are not necessarily the versions that are distributed with WebSphere Application Server.

Important: You can also determine the IBM HTTP Server version using the versionInfo command.

Procedure

1. Change the directory to the installation root of the Web server.

   For example, it is `/opt/IBM/HTTPServer` on a Solaris machine.
2. Find the subdirectory that contains the executable. The executable for IBM HTTP Server is:
   - Windows httpd.exe (the previous command, apache.exe is deprecated)
   - Linux httpd
   - AIX HP-UX Solaris apachectl
3. Issue the command with the -v option to display the version information.

   Windows
   ```
   httpd.exe -v
   ```
   Linux
   ```
   ./httpd -v
   ```
   AIX HP-UX Solaris
   ```
   ./apachectl -v
   ```

Results

The version is shown in the "Server version" field and will look something like the following:

```
IBM_HTTP_Server/7.0.0.0 (Windows)
Server built: Jul 31 2008 08:41:58
```

or

```
Server version: IBM_HTTP_Server/7.0.0.0 (Unix)
Server built: Jul 31 2008 08:41:58
```

Web server log file

Use this page to view the log file for your web server.

To view this administrative console page, click **Servers** > **Server Types** > **Web Servers** > *web_server_name* > **Log file**.

Web server log file configuration:

Access log file name	Any request that is made to the web server displays in this file.
Error log file name	Any error that occurs in the web server displays in this file.

Web server log file runtime:

Access log file name	Click **View** to display the contents of this file.
Error log file name	Click **View** to display the contents of this file.

Web server custom properties

Use this page to view and manage arbitrary name-value pairs of data, where the name is a property key and the value is a string value that can be used to set internal system configuration properties.

The administrative console contains several Custom Properties pages that work similarly. To view one of these administrative pages, click a Custom Properties link.

Web servers:

To view this administrative console page click **Servers > Server Types > Web Servers >** *web_server_name* **> Custom properties**.

Name	Specifies the name (or key) for the property.
Value	Specifies the value paired with the specified name.
Description	Provides information about the name-value pair.

Compensation service custom properties

You can specify additional settings for the compensation service through setting a custom property.

Complete the following steps to set a custom property for the compensation service.

1. Start the administrative console.
2. In the navigation pane, click **Servers** > **Server Types** > **WebSphere application servers** > ***server_name*** > **[Container Settings] Container Services** > **Compensation Service** > **[Additional Properties] Custom Properties**.
3. Click **New**.
4. On the settings page, enter the property that you want to configure in the **Name** field and the value that you want to set it to in the **Value** field.
5. Click **Apply** or **OK**.
6. Click **Save** to save your changes to the master configuration.
7. Restart the server.

You can use the custom properties page to define the following compensation service custom property:

- "Suppressing the compensation service"

Suppressing the compensation service: Not all web servers are configured to handle SOAP messages containing CoordinationContext elements. You can use WebSphere Application Server to configure a custom property for the compensation service which processes a predefined list of Enterprise Java Beans for which no CoordinationContext should be sent on web service requests.

When the compensation service is used, CoordinationContext elements are included in the outgoing SOAP header. For example:

```
<wscoor:CoordinationContext soapenv:mustUnderstand="1"
...
</wscoor:CoordinationContext>
```

If such a SOAP message is received by a web server which is not configured to process CoordinationContext elements, an exception message is produced. See the following example:

```
Header block local name 'CoordinationContext' is not defined.
```

You can construct a file containing a list of all Enterprise Java Beans which should not send the CoordinationContext element in web service requests. This file must be in plain text format and must contain one entry per line, in the following format:

```
application_name#module#bean
application_name#module#bean
application_name#module#bean
```

Here `application_name` is the name of the application as known on the server; `module` is the name of the Enterprise Java Bean jar; and `bean` is the name of the Enterprise Java Bean.

Note: This file must only contain entries for beans not configured to use the compensation service. This custom property will not be effective for any beans listed in the file which have compensation service metadata associated with them.

Name	Value
SUPPRESS_CSCOPE_ON_WS_CALLS	The fully qualified file name

Remote web server management

Use this page to configure a remote IBM HTTP Server web server.

To view the administrative console page for Remote web server management, click **Servers > Server Types > Web Servers >** *web_server_name* **> Remote web server management**.

Note: For a WebSphere Application Server, Network Deployment configuration, click **System Administration > Nodes > Add node** to create a new, unmanaged node. Then click **Servers > Web Servers > New** to create a web server using the newly-created unmanaged node.

Web servers:

Port	Indicates the port to access the administration server (default is 8008).
Use SSL	Specifies if the port is secure.
User ID	Specifies a user ID in the `<install_dir>/conf/admin.passwd` file. Create this with the `htpasswd` script file, located in the `<install_dir>/bin` directory.
Password	Specifies a password in the `<install_dir>/conf/admin.passwd` file. Create this with the `htpasswd` script file, located in the `<install_dir>/bin` directory.

Web server configuration file

Use this page to view or modify the contents of the Web server configuration file in your web browser. To view this administrative console page click **Servers > Server Types > Web Servers >** *web_server_name*. Under **Additional Properties**, click **Configuration File**.

Web servers:

If you have made changes to the configuration file you will need to restart your web server, in order for the changes to take effect.

Global directives

Use this page to configure the global directives for your web server.

To view this administrative console page, click **Servers > Server Types > Web Servers >** *web_server_name* **> Global directives**.

Security enabled:

Specifies if security is enabled in your web server.

Key store certificate alias:

If the **Security enabled** box is checked, specify the key store certificate alias.

Server name:

Specifies the hostname that the web server uses to identify itself.

Listen ports:

Specifies the port on which your web server will listen for requests.

Document root:

The directory where the web server will serve files.

Key store name:

Specifies the name you have assigned to your keystore. A button is also provided to **Manage keys and certificates**.

Target key store directory and file name:

Specifies the target directory and file name of your keystore on the machine where the web server is installed. A button is also provided to **Copy to web server key store directory**.

SSL Version 2 timeout:

Specifies the SSL Version 2 timeout.

SSL Version 3 timeout:

Specifies the SSL Version 3 timeout.

Web server virtual hosts collection

Use this page create or edit virtual hosts for your Web server.

To view this administrative console page, click **Servers > Server Types > Web Servers >** *web_server_name* **> Configuration settings > Virtual hosts**.

To create a new virtual host, click the **New** button to launch the virtual host configuration panel. To delete an existing virtual host, select the check box beside the virtual host in the list and click **Delete** .

IP address:Port:

The IP address and port number of your virtual host for the specified web server.

Server name:

Specifies the name of your virtual host for the specified web server.

Security enabled:

Specifies whether or not security is enabled for your virtual host for the specified web server. The values are **true** or **false**.

Web server virtual hosts detail

Use this page create or edit virtual hosts for your Web server.

To view this administrative console page, click **Servers > Server Types > Web Servers >** *web_server_name* **> Configuration settings > Virtual hosts > New**.

Security enabled:

Specifies whether or not security is enabled for your virtual host for the specified web server. Check the box to enable security

IP address:

The IP address of your virtual host for the specified Web server.

Port:

The port number of your virtual host for the specified web server.

Server name:

Specifies the name of your virtual host for the specified web server.

Document root:

Specifies the location of the `htdocs` directory for your web server.

Keystore filename:

Specifies the name you have assigned to your keystore.

Keystore directory:

Specifies the target directory for the key store file on the machine where your web server is installed.

Keystore certificate label:

Specifies the keystore certificate label. The certificate label specified here is the certificate that used in secure communication for this virtual host.

Editing the web server type

This topic provides information on how to change the type of Web server.

About this task

If you install a web server that is different from the one that is currently installed, you can modify the web server type from IBM HTTP Server to a non-IBM HTTP Server and vice versa, rather than delete the web server and create a new web server definition. If you change the web server type from IBM HTTP Server to non-IBM HTTP Server web server, the administration capabilities are lost accordingly.

Procedure

1. From the WebSphere Application Server administrative console, click **Servers > Server Types > Web servers**.
2. Select the server that you want to modify.
3. On the web server configuration panel, change your web server by selecting an option from the Type drop-down menu. If you are changing from a non-IBM HTTP Server to an IBM HTTP Server, you are also prompted for information such as IBM HTTP Server administration server port, user ID, and IBM HTTP Server administration server password.
4. Click **Apply**.

Results

You can verify your changes on the web servers collection panel. The web server type displays in the Web Server Type column.

Web server plug-ins

Web server plug-ins enable the web server to communicate requests for dynamic content, such as servlets, to the application server. A web server plug-in is associated with each web server definition. The configuration file (plugin-cfg.xml) that is generated for each plug-in is based on the applications that are routed through the associated web server.

A web server plug-in is used to forward HTTP requests from a supported web server to an application server. Using a web server plug-in to provide communication between a web server and an application server has the following advantages:
- XML-based configuration file
- Standard protocol recognized by firewall products
- Security using HTTPS, replacing proprietary Open Servlet Engine (OSE) over Secure Sockets Layer (SSL)

Each of the supported web server plug-ins runs on a number of operating systems. See the Supported Hardware and Software website for the product for the most current information about supported web servers. This site is located at http://www.ibm.com/support/docview.wss?rs=180&uid=swg27006921.

gotcha: The default behavior for the web server plug-in is to buffer requests up to 64 kilobytes, and retry the requests if there is no response from the application server. If you want to ensure high availability, and your HTTP requests tend to be large, set the **Maximum buffer size used when reading HTTP request content** property on the web server plug-in request routing property page in the administrative console to `-1`. Setting this property to -1 removes the maximum buffer size limit, and enables the web server plug-in to buffer all requests regardless of their size. Requests are retried if the request body fits within the buffer size. If you want to disable all request buffering, and thereby disable retries of requests with request bodies, you can set this property to `0`.

Affinity requests

Affinity requests are requests that contain a JSESSIONID. Session affinity means that all requests of the same JSESSIONID are sent to the same application server. For example, if the first request is sent to clone5, then the next affinity request from that same browser is also sent to clone5 regardless of the weight valued specified for the LoadBalanceWeight property in the plugin-cfg.xml file.

If you select `Round robin` for the **Load balancing option** Web server plug-in request routing property, and leave the IgnoreAffinityRequests property in the plugin-cfg.xml file set to its default value of `true`, the affinity requests do not lower the weight. This behavior might cause an uneven distribution of requests across the servers in environments that make use of session affinity. Setting the IgnoreAffinityRequests property to `false` causes the weight to be lowered every time an affinity request is received, which results in a more balanced round robin environment.

If you select `Random` for the **Load balancing option** property, affinity requests are still sent to the same cloneid, but new requests are routed randomly, and the value specified for the LoadBalanceWeight property is ignored.

Failover

If a request connection exceeds the time limit specified on the ConnectTimeout property in the plugin-cfg.xml file, or a 5xx error is returned from the application server, the web server plug-in marks the server as down, and attempts to connect to the next application server in the list of primary servers that is specified for the PrimaryServers property in the plugin-cfg.xml file. If the web server plug-in successfully connects to another application server, all requests that were pending for the down application server are sent to this other application server. All other new and affinity requests are sent to other servers, based on whether round robin or random is the setting for the **Load balancing option** Web server plug-in request routing property.

Failover typically does not occur the first time that the time limit specified on the ServerIOTimeout property in the plugin-cfg.xml file is exceeded for either a request or a response. Instead, the web server plug-in tries to resend the request to the same application server, using a new stream. If the time specified on the ServerIOTimeout property is exceeded a second time, the web server plug-in marks the server as down, and initiates the failover process.

gotcha: Sending a large number of pending requests to the same application server might impact the performance of that application server if a failover situation occurs. You can use the MaxConnections property to limit the number of requests that might be pending for an application server.

Running multiple web server child processes

You can configure most web servers to start multiple child processes. In this situation, each child process loads its own instance of the web server. When running multiple web server child processes, remember that:

- Multiple running instances of the web server plug-in cannot share information. Therefore the dynamically changing load balance weight of each application server is not shared between the web server plug-in instances. For example, one instance of the web server plug-in might consider an application server to be running with a weight of 5, while another instance of the web server plug-in might be consider the same application server to be down and unusable. This difference in perspective might cause an incoming request to be handled differently, depending on which web server plug-in instance handles the request.
- The web server plug-in settings are handled on a per instance basis. For example, the MaxConnections property specifies the number of pending requests that are allowed on that web server, for each web server plug-in instance. If the MaxConnections property is set to `20`, and you start three web server child processes, each of the three web server plug-in instances allow 20 pending connections to the same application server, which means that there could be up to 60 pending connections.

Selecting a front end for your WebSphere Application Server topology

You can use either a web server plug-in, a WebSphere Application Server proxy server, or a DMZ Secure Proxy Server for IBM WebSphere Application Server to provide session affinity, failover support, and workload balancing for your WebSphere Application Server topology.

After you install the product, you can set up either a web server plug-in, a WebSphere Application Server proxy server, or a DMZ Secure Proxy Server for IBM WebSphere Application Server to establish communication between an application and a remote client.

- A WebSphere Application Server web server plug-in provides an interface between a web server and an application server. The web server plug-in determines the server to which a client request for dynamic content, such as servlets, needs to be routed.
- A WebSphere Application Server proxy server is a specific type of application server that routes HTTP requests to content servers that perform the work. The WebSphere Application Server proxy server can be the initial point of entry for requests to servers in your enterprise environment. However, because a WebSphere Application Server proxy server is not safe for DMZ deployment, a WebSphere Application Server proxy server is typically fronted by a web server, or used in internal only environments where stringent host security requirements are not required.

- A DMZ Secure Proxy Server for IBM WebSphere Application Server is a WebSphere Application Server proxy server that is designed specifically to be safely installed on a stand-alone node in a demilitarized zone (DMZ). If you require the function of the WebSphere Application Server proxy server, and want to deploy it to the DMZ, you should use a DMZ Secure Proxy Server for IBM WebSphere Application Server to provide session affinity, failover support, and workload balancing for your WebSphere Application Server topology.

  The DMZ is a safe zone between firewalls that is typically located between a client and a backend server. A DMZ Secure Proxy Server for IBM WebSphere Application Server accepts requests from clients on the Internet, and forwards the requests to servers in your enterprise environment.

You can also use the on demand router (ODR), that is provided with the WebSphere eXtended Deployment product, as a reverse proxy between an HTTP client and a clustered application, or a partitioned application. See the WebSphere eXtended Deployment Information Center at http://publib.boulder.ibm.com/infocenter/wxsinfo/v7r0/index.jsp for more information about using ODR.

The traditional topology, using your web server of choice and the corresponding web server plug-in, is recommended unless:
- You want to use specific features of the ODR or a WebSphere Application Server proxy server.
- You want to use another function, such as Session Initiation Protocol (SIP) or WS-Addressing affinity/failover, which requires a WebSphere Application Server proxy server.

The following tables compare the core application server frontend functionality, and the non-core functionality of a web server plug-in running in a modern web server, such as the IBM HTTP Server, based on Apache HTTP Server, a WebSphere Application Server proxy server, and a DMZ Secure Proxy Server for IBM WebSphere Application Server.

Table 4. Core functionality. This table compares the functionality that a web server plug-in, a WebSphere Application Server proxy server, and a DMZ Secure Proxy Server for IBM WebSphere Application Server provide.

Functionality	**Web server plug-in used with either the IBM HTTP Server or the Apache Web Server**	**WebSphere Application Server proxy server**	**DMZ Secure Proxy Server for IBM WebSphere Application Server**
Session affinity	Yes	Yes[1, 2]	Yes[1, 2]
DMZ ready	Yes	No	Yes
Custom advisors are supported	No	Yes	Yes
Service Level Agreement (SLA)	No	No	No
QoS/Throttling	No	No	No
SIP proxy	No	Yes	Yes
ESI dynamic Caching	Yes	Yes[3]	Yes[3]
Managed from the administrative console	Yes	Yes	Yes[4]
Stream caching (large response caching)	Yes	Yes	Yes
Dynamically receive management events[5]	No	Yes	Yes[6]
Multi cells routing	No	Yes[7]	Yes[8]
Performance monitoring	Yes[9]	Yes[10]	Yes[10]
Load Balancing (weighted round-robin)	Yes[11]	Yes[11]	Yes[11]

Table 4. Core functionality (continued). This table compares the functionality that a web server plug-in, a WebSphere Application Server proxy server, and a DMZ Secure Proxy Server for IBM WebSphere Application Server provide.

Functionality	Web server plug-in used with either the IBM HTTP Server or the Apache Web Server	WebSphere Application Server proxy server	DMZ Secure Proxy Server for IBM WebSphere Application Server
Routing rules are configurable	No[12]	Yes	Yes
Interoperability with WLM	Yes[13]	Yes[14]	Yes
Web service affinity and failover	No	Yes[15]	No
Rule expression and custom routing	No	Yes[16]	Yes[15]
Generic server cluster (GSC) affinity and failover	No	Yes[17]	Yes[16]

Table notes:

1. Session affinity is supported for WebSphere Application Server managed resources. However, some session management custom properties, such as HttpSessionCloneId, are not supported.
2. For generic server routing, where the resources are not WebSphere Application Server managed resources, active session affinity and passive session affinity need to be configured under generic server routing action.
3. WebSphere Application Server proxy servers and DMZ Secure Proxy Servers for IBM WebSphere Application Server do not support fragment caching. Only whole page caching, and the ESI invalidation servlet are supported.
4. Secure proxy profile on a DMZ installation can only be managed using scripting or an administrative agent. Configuration-only secure proxy profile can be managed through scripting or the administrative agent console. If you use an administrative agent console, you must register a proxy profile with the administrative agent.
5. As performed by ODR in a WebSphere Extended Deployment environment.
6. Static routing needs to be turned off and core group bridge tunneling needs to be enabled for both the DMZ Secure Proxy Server for IBM WebSphere Application Server, and the core group bridge interface for the WebSphere Application Server, Network Deployment cells.
7. Requires core group bridge setup between the proxy cell and other cells.
8. Static routing needs to be turned off and core group bridge tunneling needs to be enabled for both the DMZ Secure Proxy Server for IBM WebSphere Application Server, and the core group bridge interface for the WebSphere Application Server, Network Deployment cells.
9. The web server plug-in statistics are obtained from request metrics.
10. WebSphere Application Server proxy server statistics and DMZ Secure Proxy Server for IBM WebSphere Application Server statistics can be retrieved from Tivoli performance viewer, ARM, and performance mBeans.
11. Random Load balancing is supported in addition to weighted round robin.
12. Web server plug-in can only do static routing.
13. A web server plug-in indirectly has interoperability with WLM through the exchange of dynamic workload manager (DWLM) Partition Tables between the web server plug-in and WebSphere Application Server. The plug-in uses these tables for dynamic routing and failover scenarios within a cluster.
14. The proxy server uses the WebSphere Application Server WLM even if the proxy server is running on a z/OS operating system.
15. The DataPower appliance manager provides faster web service affinity and failover service than Java proxy provides.
16. Rule expression and custom routing allows administrators to override default WebSphere Application Server routing behavior. For example, you might not want requests forwarded to server1 in a cluster between 11:00 PM and 12:00 PM because you regularly apply maintenance to that server during that time interval.
17. Proxy server supports load balancing and failover for generic server clusters with passive and/or active affinity.

Table 5. Functionality provided outside of the web server plug-in. This table provides a comparison of the functionality that a typical web server, that is hosting a web server plug-in, a WebSphere Application Server proxy server, and a DMZ Secure Proxy Server for IBM WebSphere Application Server provide outside of the core application server frontend functionality. See your web server documentation for a complete description of the functionality that your particular web server provides.

Functionality	**Web server plug-in used with either the IBM HTTP Server or the Apache Web Server**	**WebSphere Application Server proxy server**	**DMZ Secure Proxy Server for IBM WebSphere Application Server**
Common Gateway Interface (CGI)	Yes	No	No
Request URI rewriting	Yes	No	No
Efficient static file serving	Yes	Basic[1]	Basic[1]
Compression	Yes	Yes	Yes
Response filtering	Yes	Yes[2]	Yes[2]
SSL termination	Yes	Yes	Yes
Cryptographic Accelerator[3]	Yes	Yes[4]	Yes[4]
FIPS	Yes	Yes	Yes
Third-party/customer-written plug-ins	Yes	No	No
Logging	Yes	Yes[5]	Yes[5]
Custom Logging	Yes	No	No
Disk caching	Yes	Yes	Yes
Asynchronous request handling	none or partial[6]	Yes[7]	

Table notes:

1. WebSphere Application Server proxy servers support basic static file serving.
2. WebSphere Application Server proxy servers support HTML link rewriting.
3. This functionality only applies to Cryptographic Accelerators that WebSphere Application Server supports. See the Supported hardware and software web page .
4. The support is provided by IBM JDK/JCE.
5. Only NCSA common format is supported.
6. The connection between a web server plug-in and an application server is synchronous and consumes a thread while reading/writing or waiting for data. See your web server documentation for information about how your particular web server handles client connections.
7. Proxy server is optimized to handle AJAX long polling requests under large scale deployments.

Web server plug-in connections

The web server plug-ins are used to establish and maintain persistent HTTP and HTTPS connections to application servers.

When the plug-in is ready to send a request to the application server, it first checks its connection pool for existing connections. If an existing connection is available the plug-in checks its connection status. If the status is still good, the plug-in uses that connection to send the request. If a connection does not exist, the plug-in creates one. If a connection exists but has been closed by the application server, the plug-in closes that connection and opens a new one.

After a connection is established between a plug-in and an application server, it will not be closed unless the application server closes it for one of the following reasons:

- If the **Use Keep-Alive** property is selected and the time limit specified on the **Read timeout** or **Write timeout** property for the HTTP inbound channel has expired.
- The maximum number of persistent requests which can be processed on an HTTP inbound channel has been exceeded. This number is set using the **Maximum persistent requests** property that is specified for the HTTP inbound channel.
- The Application Server is shutting down.

Even if the application server closes a connection, the plug-in will not know that it has been closed until it tries to use it again. The connection will be closed if one of the following events occur:
- The plug-in receives a new HTTP request and tries to reuse the existing connection.
- The number of httpd processes drop because the web server is not receiving any new HTTP requests. For the IBM HTTP Server, the number of httpd processes that are kept alive depends on the value specified on the web server's MinSpareServers directive.
- The web server is stopped and all httpd processes are terminated, and their corresponding sockets are closed.

gotcha: Sometimes, if a heavy request load is stopped or decreased abruptly on a particular application server, a lot of the plug-in's connections to that application server will be in CLOSE_WAIT state. Because these connections will be closed the first time the plug-in tries to reuse them, having a large number of connections in CLOSE-WAIT state should not affect performance

Web server plug-in remote user information processing

You can configure your web server with a vendor-acquired authentication module and then configure the web server plug-in to route requests to an application server.

If an application calls the getRemoteUser method, it relies on a private HTTP header that contains the remote user information and is parsed by the plug-in. The plug-in sets the private HTTP header value whenever a web server authentication module populates the remote user in the web server data structure. If the private HTTP header value is not set, the call to getRemoteUser method by the application returns a null value.
- In the case of an Apache Web Server or the IBM HTTP Server, the plug-in builds the private header from the information contained in the associated request record.
- In the case of a Sun One Web Server, the plug-in builds the private header from the information contained in the **auth_user** property associated with the request. The private header is usually set to the name of the local HTTP user of the web browser, if HTTP access authorization is activated for the URL.
- In the case of a Domino Web Server, the plug-in builds the private header from the information contained in the **REMOTE_USER** environment variable. The plug-in sets this variable to **anonymous** for users who have not logged in and to the *username* for users who are logged into the application.
- In the case of an Internet Information Services (IIS) Web Server, the plug-in builds the private header from the information contained in the **REMOTE_USER** environment variable. The plug-in sets this variable to the name of the user as it is derived from the authorization header sent by the client.

Note: If the private header is not being set in the Sun One, IIS, or Domino Web Server plug-in, make sure the request record includes information about the user requesting the data.

Note: If an call to getRemoteUser method by the application returns a null value, or if the correct remote user information is not being added to the data structure for the web server plug-in, make sure the remote user parameter within the vendor-acquired authentication module is still set to **YES**. (Sometimes this parameter gets set to **NO** when service is applied.)

Private headers

A web server plug-in can use private headers to forward requests for dynamic content, such as servlets, to the application server.

After you configure a web server plug-in, in addition to regular plug-in functions, you can use private headers as a mechanism for forwarding proxy information from the plug-in to an application server. This information is not normally included in HTTP requests.

Private headers are implemented as a set of HTTP request header name and value pairs that the plug-in adds to the HTTP request header before the request is forwarded to an application server. The application server's web container removes this information from the header and then processes this information.

Private headers can include such information as the remote (client) user, the remote (client) host name, or an SSL client certificate. They conform to a naming standard so that there is no namespace collision with the architected HTTP header fields.

For example, authentication information, such as a client certificate, is normally requested by the web server once during the establishment of an HTTP session. It is not required again for individual requests within that session. However, a client certificate must accompany each request forwarded to the application server. The application server can then use the certificate as needed.

Similarly, the web server examines TCP/IP socket connections for information about the host address of the client. The application server cannot perform this examination because its socket connection is with the plug-in and not with the actual client. Therefore, one of the private headers is the host address of the actual client.

Gskit install images files

The Global Security Kit (GSKit) installation image files for the WebSphere Web server plug-ins are packaged on the CD with the web server plug-in files.

You can download the appropriate GSKIT file to the workstation on which your web server is running.

The Global Security Kit (GSKit) installation images files for the WebSphere Web Server Plug-ins are provided as a "local install" and are not installed globally nor are registered with the native package management utilities.

Plug-ins: Resources for learning

Use the following links to find relevant supplemental information about web server plug-ins. The information resides on IBM and non-IBM Internet sites, whose sponsors control the technical accuracy of the information.

These links are provided for convenience. Often, the information is not specific to the IBM WebSphere Application Server product, but is useful all or in part for understanding the product. When possible, links are provided to technical papers and Redbooks® that supplement the broad coverage of the release documentation with in-depth examinations of particular product areas.

View links to additional information about:
- Programming model and decisions
- Programming instructions and examples

Programming model and decisions

- Best Practice: WebSphere Plug-in Configuration Regeneration at http://www-128.ibm.com/developerworks/websphere/library/bestpractices/plug_in_configuration_regeneration.html

Programming instructions and examples

- WebSphere Application Server education at http://www-128.ibm.com/developerworks/search/searchResults.jsp?searchType=1&searchSite=dW&searchScope=dW&query=WebSphere+education
- Listing of all IBM WebSphere Application Server Redbooks at http://publib-b.boulder.ibm.com/Redbooks.nsf/Portals/WebSphere

Installing and configuring web server plug-ins

WebSphere Application Server supplies a unique binary plug-in module for each supported web server. The plug-in configuration file, which the WebSphere Application Server products create and maintain, interacts with the binary module to provide information about the application server configuration to the web server. The web server uses the information to determine how to communicate with the application server.

Before you begin

trns: In WebSphere Application Server Version 7 and earlier, you use the Plug-ins installation wizard to install the plug-in module, configure the web server for communicating with the application server, and create a web server configuration definition in the application server if possible. In WebSphere Application Server Version 8 and later, you must first install the Web Server Plug-ins, the web server, and the WebSphere Customization Toolbox; then, you run the Web Server Plug-ins Configuration Tool that is contained in the toolbox to configure the web server to communicate with the application server and, if possible, create a web server configuration definition in the application server. If you know how to set up your initial web server and plug-in configuration manually, you can do so in Version 7 and Version 8.

Go to http://www-1.ibm.com/support/docview.wss?rs=180&context=SSEQTP&uid=swg21160581 for information about how to verify what Version 4.0 through Version 8.0 plug-in versions are installed on local or remote web servers and how to determine if the installation complies with supported configurations.

You must install a supported web server before you can install and configure a plug-in for the web server.

The Web Server Plug-ins Configuration Tool configures the web server for communicating with the application server and creates a web server configuration definition in the application server if possible.

Some topologies, such as the web server on one system and the application server on another system, prevent the Web Server Plug-ins Configuration Tool from creating the web server definition in the application server configuration directly on the remote system. In such a case, the Web Server Plug-ins Configuration Tool creates a script that you can copy to the application server system. Run the script to create the web server configuration definition within the application server configuration.

About this task

This article describes installing and configuring web server plug-ins for WebSphere Application Server. WebSphere Application Server products supplies a unique binary plug-in module for each supported web server. The plug-in configuration file, which the WebSphere Application Server products create and maintain, interacts with the binary module to provide information about the application server configuration to the web server. The web server uses the information to determine how to communicate with the application server.

The Web Server Plug-ins Configuration Tool configures the web server and the application server to allow communication between the servers.

Tip: As an alternative to using the Web Server Plug-ins Configuration Tool, you can use the pct command-line tool with a response file to configure a web server. Read "Configuring a web server plug-in using the WCT command-line utility" on page 132 for more information.

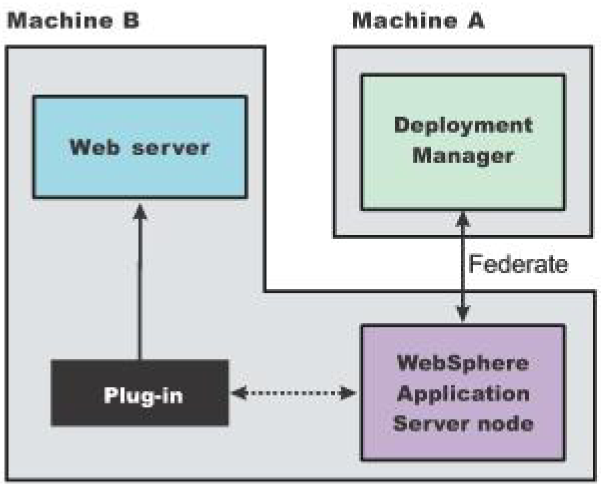
Select one of the following topology scenarios and follow the steps below the diagram to install the plug-in and configure both the web server and the application server.

When multiple profiles exist, you can select the profile that the Web Server Plug-ins Configuration Tool configures. See “Plug-ins configuration” on page 91 for a description of the flow of logic that determines how to select the profile to configure.

Procedure

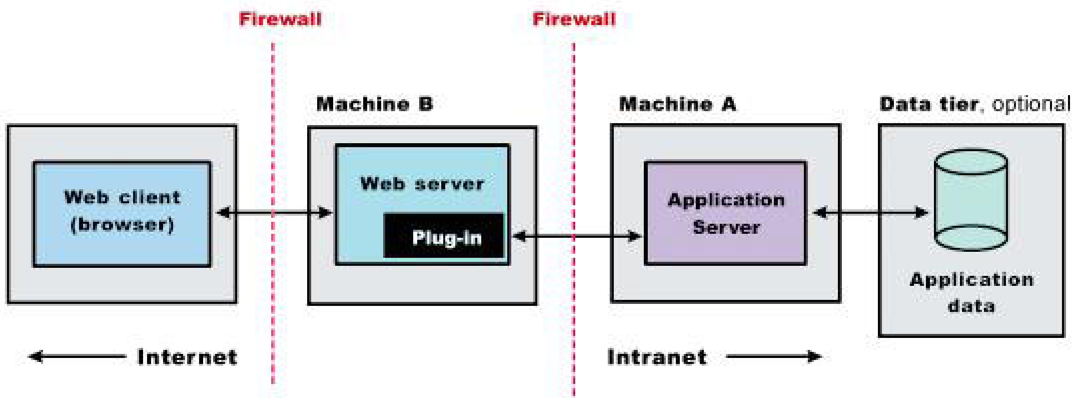
- **Scenario 1: Local application server profile** The application server and the web server are on a single system or logical partition.

  A local distributed installation includes the Web server plug-in, the web server, and a *managed* application server on the same system:



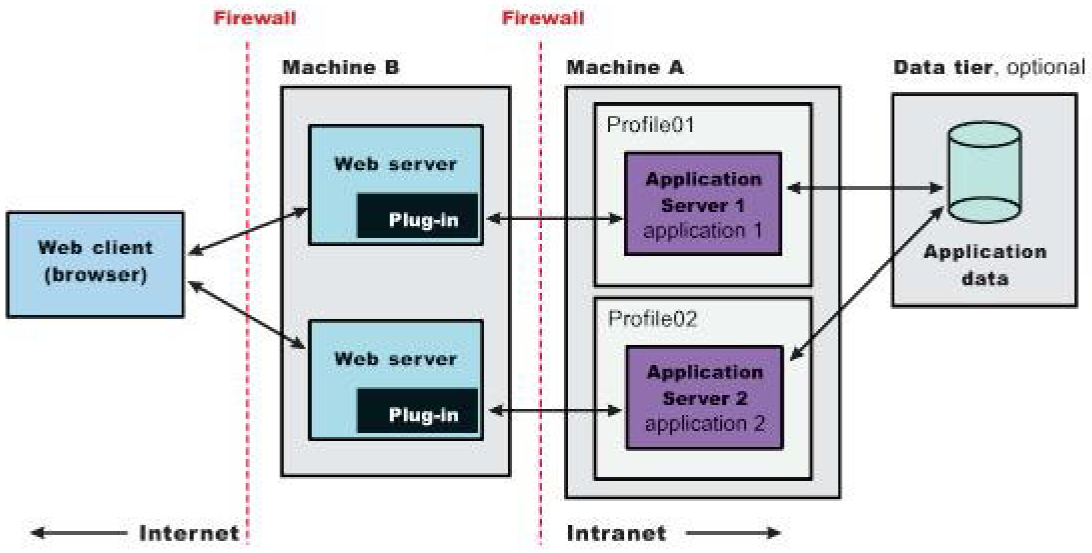
  See “Configuring a web server and an application server profile on the same machine” on page 98 for the procedure that explains how to create this web server topology for an application server profile.

- **Scenario 2: Remote** The application server and the web server are on separate machines or logical partitions.



  See “Configuring a web server and an application server on separate machines (remote)” on page 107 for the procedure that explains how to create this web server topology.

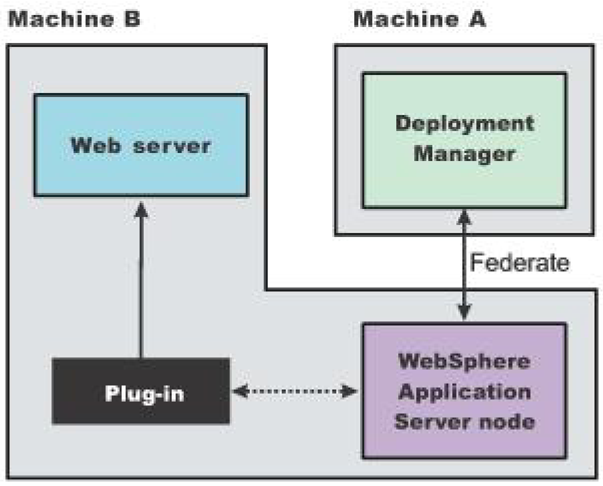
- **Scenario 3: Remote** Multiple standalone application servers are on one system, and each application server has a dedicated web server on a separate system or logical partition.



See “Configuring multiple web servers and remote standalone application servers” on page 115 for the procedure that explains how to create this web server topology.

- **Scenario 4: Local custom profile** A managed node and the web server are on the same system or logical partition.

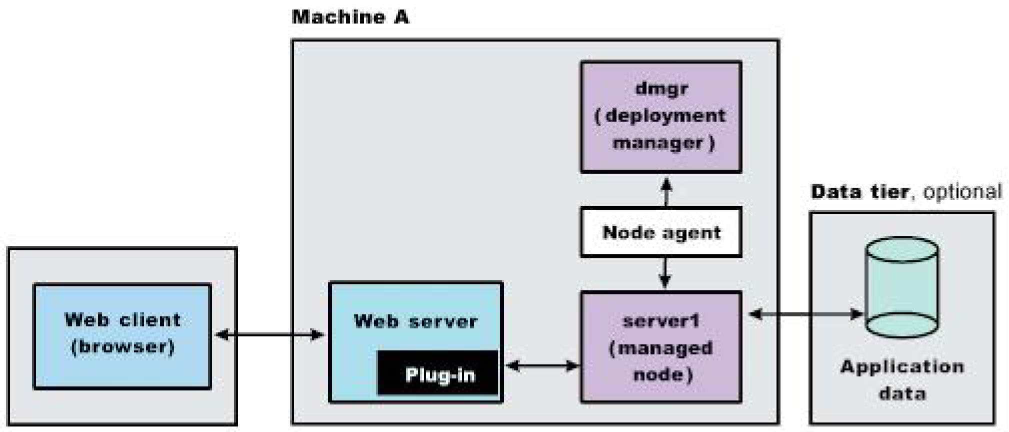
  A local distributed installation includes the web server plug-in, the web server, and the managed custom node on the same system:



  See “Configuring a web server and a custom profile on the same machine” on page 122 for the procedure that explains how to create this web server topology for a federated custom profile.

- **Scenario 5: Local deployment manager profile** A deployment manager node and the web server are on a single system or logical partition.

  A local distributed installation includes the web server plug-in, the web server, and the application server on the same system:



See “Configuring a web server and a deployment manager profile on the same machine” on page 128 for the procedure that explains how to create this web server topology for a deployment manager profile.

Results

You can install a web server and the web server plug-ins for various standalone application server topologies by following the procedures described in this article.

What to do next

See “Selecting a web server topology diagram and roadmap” for an overview of the installation procedure.

See “Web server configuration” on page 39 for more information about the files involved in configuring a web server.

See Editing Web server configuration files for information about how the Web Server Plug-ins Configuration Tool configures supported web servers.

Selecting a web server topology diagram and roadmap

Install and configure the Web Server Plug-ins to allow the application server to communicate with the web server.

Before you begin

The primary production configuration for a web server is an application server on one machine and a web server on a separate machine. This configuration is referred to as a *remote* configuration. Contrast the remote configuration to the local configuration, where the application server and the web server are on the same machine.

About this task

The Web Server Plug-ins Configuration Tool has three main tasks:

- Configures the web server configuration file on the web server machine to point to the binary plug-in module and to the XML configuration file for the binary module.
- Installs a temporary XML configuration file for the binary module (`plugin-cfg.xml`) on the web server machine in remote scenarios.

- Creates the configuration for a web server definition on the application server machine. The Web Server Plug-ins Configuration Tool processes the creation of the web server definition differently depending on the scenario:

  Web server plug-in installation for standalone application server environments

  – Recommended remote standalone application server installation:

    Creates a configuration script that you run on the application server machine. Install the web server and configure its plug-in on a different machine than the application server. This configuration is recommended for a production environment.

  – Local standalone application server installation:

    Configures the default profile on a local application server machine and creates the Web server definition for it directly. Install the web server and configure its plug-in on the same machine with the application server. This configuration is for development and test environments.

  Web server plug-in installation for distributed environments (cells)

  – Recommended remote distributed installation:

    Creates a configuration script that you run on the application server machine. Install the web server and configure its plug-in on a different machine than the deployment manager or managed node. This configuration is recommended for a production environment.

  – Local distributed installation:

    Creates a configuration script that you run when the deployment manager is running. Install the web server and configure its plug-in on the same machine with the deployment manager or a managed node. This configuration is for development and test environments.

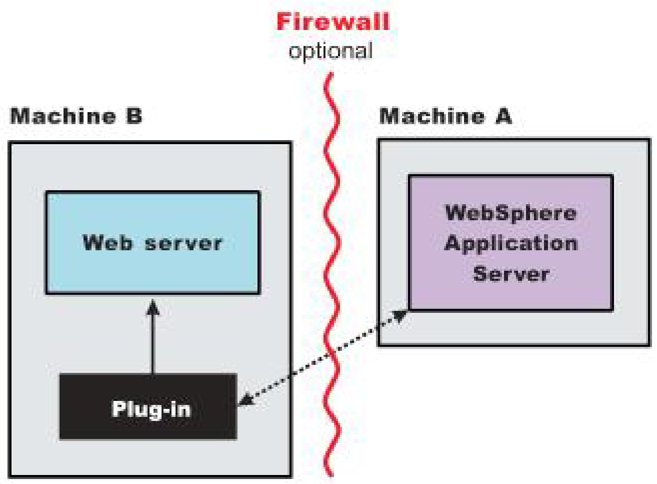
Tip: As an alternative to using the Web Server Plug-ins Configuration Tool, you can use the pct command-line tool with a response file to configure a web server. Read “Configuring a web server plug-in using the WCT command-line utility” on page 132 for more information.

Procedure

- **Set up a remote web server installation.**

  The remote web server configuration is recommended for production environments.

  The remote installation installs the web server plug-in on the web server machine when the application server is on a separate machine, such as shown in the following graphic:



Remote installation scenario

Table 6. Installation and configuration. Remote installation scenario

Step	Machine	Task
1	A	Install Installation Manager.
2	A	Use Installation Manager to install the WebSphere Application Server product.
3	A	Create a standalone application server profile.
4	B	Install Installation Manager.
5	B	Use Installation Manager to install the following: • Web Server Plug-ins for WebSphere Application Server • WebSphere Customization Toolbox
6	B	Use Installation Manager to install IBM HTTP Server, or install another supported web server.
7	B	Open the WebSphere Customization Toolbox, and launch the Web Server Plug-ins Configuration Tool to configure the web server plug-in. The script for creating and configuring the web server is created under the *plugins_root*/`bin` directory.
8	B	Copy the `configure`*`web_server_name`* script to paste on Machine A. If one machine is running under an operating system such as AIX or Linux and the other machine is running under Windows, copy the script from the *plugins_root*/`bin/crossPlatformScripts` directory.
9	A	Paste the `configure`*`web_server_name`* script from Machine B to the *app_server_root*/`bin` directory on Machine A.
10	A	Start the application server.
11	A	Run the script from a command line.
12	A	Verify that the application server is running. Open the administrative console and save the changed configuration.
13	B	Start the web server. AIX HP-UX Linux Solaris Source the *plugins_root*/`setupPluginCfg.sh` script for a Domino Web Server before starting a Domino Web Server.
14	B	Run the Snoop servlet. Access the following URL in your browser: `http://`*`host_name_of_machine_B`*`:`*`http_transport_port`*`/Snoop` To verify with your own application, regenerate and propagate the `plugin-cfg.xml` file after installing the application.

Regeneration of the `plugin-cfg.xml` file

The web server plug-in configuration service regenerates the `plugin-cfg.xml` file automatically.

During configuration, the temporary `plugin-cfg.xml` file is installed on Machine B in the *plugins_root*/`config/`*`web_server_name`* directory. To use the actual `plugin-cfg.xml` file from the application server, propagate the `plugin-cfg.xml` file as described in the next section.

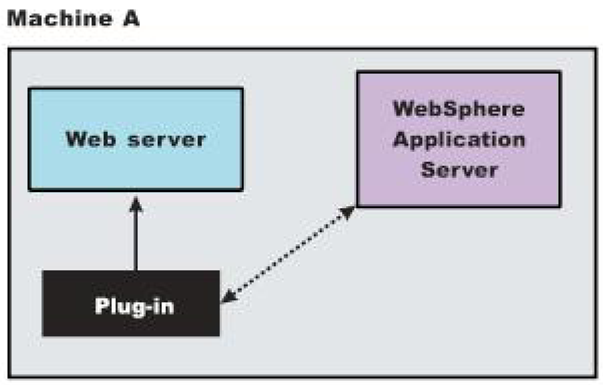
Propagation of the `plugin-cfg.xml` file

The web server plug-in configuration service propagates the `plugin-cfg.xml` file automatically for IBM HTTP Server Version 6.0 or later. For all other web servers, propagate the plug-in configuration file manually. Copy the `plugin-cfg.xml` file from the *profile_root*/ `config/cells/`*`cell_name`*`/nodes/`*`web_server_name`*`_node/servers/`*`web_server_name`* directory on Machine A. Paste the file into the *plugins_root*/`config/`*`web_server_name`* directory on Machine B.

- **Set up a local web server configuration.**

  The local web server configuration is recommended for a development or test environment.

  A local installation includes the web server plug-in, the web server, and the Application Server on the same machine:



Local installation scenario

Table 7. Installation and configuration. Local installation scenario

Step	Machine	Task
1	A	Install Installation Manager.
2	A	Use Installation Manager to install the following: • WebSphere Application Server product • Web Server Plug-ins for WebSphere Application Server • WebSphere Customization Toolbox
3	A	Use Installation Manager to install IBM HTTP Server, or install another supported web server.
4	A	Create a standalone application server profile.
5	A	Open the WebSphere Customization Toolbox, and launch the Web Server Plug-ins Configuration Tool to configure the web server plug-in and create the web server definition. The web server definition is automatically created and configured during the configuration of the plug-in.
6	A	Start the application server.
7	A	Verify that the application server is running. Open the administrative console and save the changed configuration.
8	A	Start the web server. AIX HP-UX Linux Solaris Run the *plugins_root*/`setupPluginCfg.sh` script for a Domino Web Server before starting a Domino Web Server.
9	A	Run the Snoop servlet. Access the following URL in your browser: `http://`*`host_name_of_machine_A`*`:`*`http_transport_port`*`/Snoop` To verify with your own application, regenerate and propagate the `plugin-cfg.xml` file after installing the application.

Regeneration of the `plugin-cfg.xml` file

The web server plug-in configuration service regenerates the `plugin-cfg.xml` file automatically.

The `plugin-cfg.xml` file is generated in the *profile_root*/`config/cells/`*`cell_name`*`/nodes/`*`web_server_name`*`_node/servers/`*`web_server_name`* directory. The generation occurs when the web server definition is created.

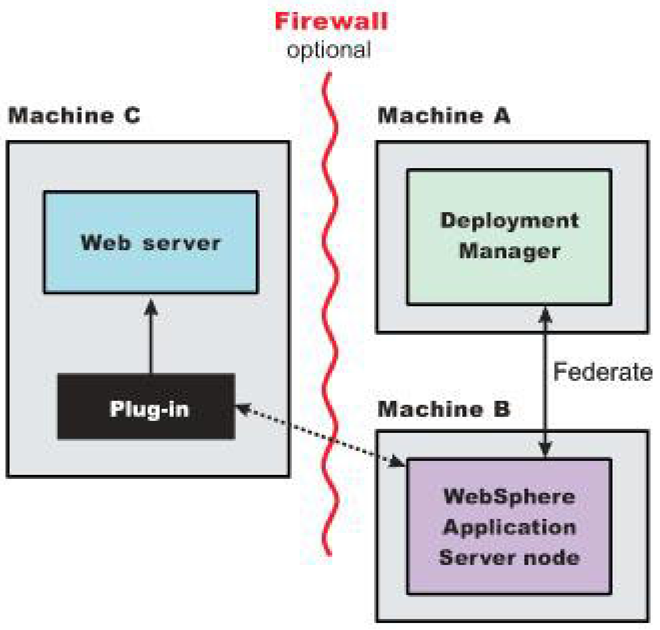
Propagation of the `plugin-cfg.xml` file

The local file does not require propagation.

- **Set up a remote web server installation in a cell.**

  The remote web server configuration is recommended for production environments.

  The remote distributed installation installs the web server plug-in on the web server machine when the application server is on a separate machine, such as shown in the following graphic:



Remote distributed installation scenario

Table 8. Installation and configuration. Remote distributed installation scenario

Step	Machine	Task
1	A	Install IBM Installation Manager.
2	A	Use Installation Manager to install the WebSphere Application Server product.
3	A	Create a deployment manager profile.
4	A	Verify that the deployment manager is running to allow node synchronization of changed configuration files.
5	B	Install Installation Manager.
6	B	Use Installation Manager to install the WebSphere Application Server product.
7	B	Create a standalone application server.
8	B	Add the node into the deployment manager cell in order to start the node agent process. Start the node agent on an existing managed node. The deployment manager and the node agent must be running to allow node synchronization of changed configuration files.
9	C	Install Installation Manager.
10	C	Use Installation Manager to install the following: • Web Server Plug-ins for WebSphere Application Server • WebSphere Customization Toolbox
11	C	Use Installation Manager to install IBM HTTP Server, or install another supported web server.
12	C	Open the WebSphere Customization Toolbox, and launch the Web Server Plug-ins Configuration Tool to configure the web server plug-in. The script for creating and configuring the web server is created under the *plugins_root*/`bin` directory.
13	C	Copy the `configure`*`web_server_name`* script to paste on Machine A. If one machine is running under an operating system such as AIX or Linux and the other machine is running under Windows, copy the script from the *plugins_root*/`bin/crossPlatformScripts` directory.
14	A	Paste the `configure`*`web_server_name`* script from Machine C to the *app_server_root*/`bin` directory on Machine A.
15	A	Start the node agent and the deployment manager if they are not already running, then run the script from a command line. If you have enabled security or changed the default JMX connector type, edit the script and include the appropriate parameters on the wsadmin command.
16	A and B	Use the administrative console of the deployment manager on Machine A to start the application server on Machine B. Wait for synchronization to occur and save the new configuration.

Table 8. Installation and configuration (continued). Remote distributed installation scenario

Step	Machine	Task
17	C	Start the web server. AIX HP-UX Linux Solaris Run the *plugins_root*/`setupPluginCfg.sh` script for a Domino Web Server before starting a Domino Web Server.
18	C	Run the Snoop servlet. Access the following URL in your browser: `http://`*`host_name_of_machine_C`*`:`*`http_transport_port`*`/Snoop` To verify with your own application, regenerate and propagate the `plugin-cfg.xml` file after installing the application.

Regeneration of the `plugin-cfg.xml` file

The web server plug-in configuration service regenerates the `plugin-cfg.xml` file automatically.

During configuration, the temporary `plugin-cfg.xml` file is installed on Machine B in the *plugins_root*/`config`/*`web_server_name`* directory.

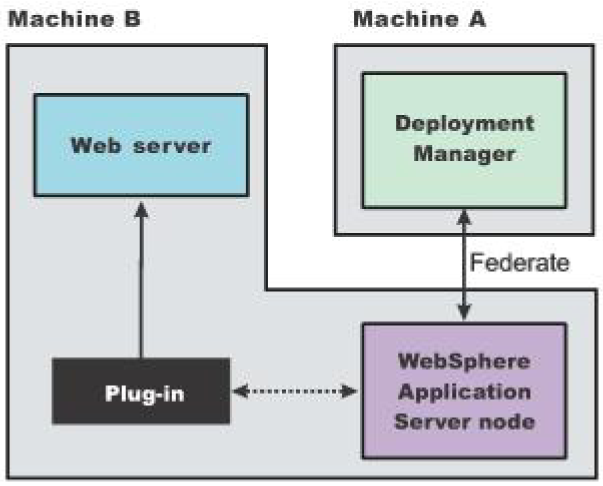
Propagation of the `plugin-cfg.xml` file

The web server plug-in configuration service propagates the `plugin-cfg.xml` file automatically for IBM HTTP Server Version 6.0 or later. For all other web servers, propagate the plug-in configuration file, by manually copying the `plugin-cfg.xml` file from the *profile_root*/`config/cells/`*`cell_name`*`/nodes/`*`node_name`*`/servers/`*`web_server_name`* directory on Machine A to the *plugins_root*/`config/`*`web_server_name`* directory on Machine C.

- **Set up a local distributed web server configuration.**

  The local web server configuration is recommended for a development or test environment.

  A local distributed installation includes the web server plug-in, the Web server, and the managed application server on the same machine:



Local distributed installation scenario

Table 9. Installation and configuration. Local distributed installation scenario

Step	Machine	Task
1	A	Install IBM Installation Manager.
2	A	Use Installation Manager to install the WebSphere Application Server product.
3	A	Create a deployment manager profile.
4	A	Verify that the deployment manager is running to allow node synchronization of changed configuration files.
5	B	Install Installation Manager.

Table 9. Installation and configuration (continued). Local distributed installation scenario

Step	Machine	Task
6	B	Use Installation Manager to install the following: • WebSphere Application Server product • Web Server Plug-ins for WebSphere Application Server • WebSphere Customization Toolbox
7	B	Use Installation Manager to install IBM HTTP Server, or install another supported web server.
8	B	Create a standalone application server.
9	B	Add the node into the deployment manager cell in order to start the node agent process. Start the node agent on an existing managed node. The deployment manager and the node agent must be running to allow node synchronization of changed configuration files.
10	B	Open the WebSphere Customization Toolbox, and launch the Web Server Plug-ins Configuration Tool to configure the web server plug-in and create the web server definition. The script for creating and configuring the web server is created in the *plugins_root*/`bin` directory.
11	B	Copy the `configure`*`web_server_name`* script to paste on Machine A.
12	A	Paste the `configure`*`web_server_name`* script from Machine B to the *`app_server_root`*`/bin` directory on Machine A.
13	A	After verifying that the deployment manager and the node agent are running on Machine A, run the `configure`*`web_server_name`* script from a command line in the *plugins_root*/`bin` directory. If you have enabled security or changed the default JMX connector type, edit the script and include the appropriate parameters.
14	A and B	Use the administrative console of the deployment manager on Machine A to start the application server on Machine B. Wait for synchronization to occur and save the new configuration.
15	B	Start the web server. AIX HP-UX Linux Solaris Run the *plugins_root*/`setupPluginCfg.sh` script for a Domino Web Server before starting a Domino Web Server.
16	B	Run the Snoop servlet. Access the following URL in your browser: `http://`*`host_name_of_machine_B`*`:`*`http_transport_port`*`/Snoop`

Regeneration of the `plugin-cfg.xml` file

The web server plug-in configuration service regenerates the `plugin-cfg.xml` file automatically.

During configuration, the temporary `plugin-cfg.xml` file is installed on Machine B in the *plugins_root*/`config/`*`web_server_name`* directory.

The `plugin-cfg.xml` file is generated at the location *profile_root*/`config/cells/`*`cell_name`*`/nodes/`*`node_name`*`/servers/`*`webServerName`* directory, when the web server definition is created.

Regenerate the `plugin-cfg.xml` file in the web server definition in the application server whenever the configuration changes. The web server has immediate access to the file whenever it is regenerated.

When the web server plug-in configuration service (an administration service) is enabled on Machine A, the `plugin-cfg.xml` file is automatically generated for all web servers.

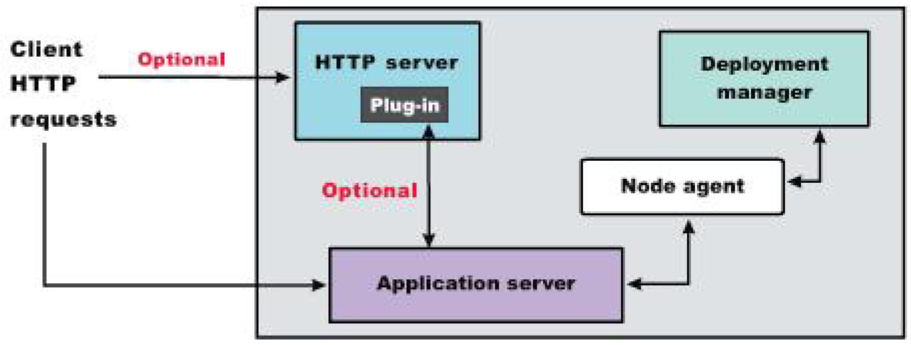
Propagation of the plugin-cfg.xml file

Node synchronization is used to propagate the `plugin-cfg.xml` file from Machine A to Machine B.

When the web server plug-in configuration service (an administration service) is enabled on Machine A, the `plugin-cfg.xml` file is automatically propagated for all web servers.

Alternate configuration

This procedure describes installing the plug-ins on two machines. However, you can perform this procedure on a single machine as shown in the following graphic. A local distributed installation also includes the web server plug-in, the web server, the application server, and the deployment manager on the same machine:



Results

You can set up a remote or local web server by installing Application Server, the web server, and then the Web Server Plug-ins.

What to do next

See “Web server configuration” on page 39 for more information about the files involved in configuring a web server.

See “Plug-ins configuration” on page 91 for information about the logic behind the processing scenarios for the Web Server Plug-ins Configuration Tool.

See “Editing web server configuration files” on page 12 for information about how the Web Server Plug-ins Configuration Tool configures supported web servers.

See “Installing and configuring web server plug-ins” on page 58 for information about other installation scenarios for installing Web Server Plug-ins.

Installing and uninstalling the Web Server Plug-ins on distributed operating systems

IBM Installation Manager is a common installer for many IBM software products that you use to install, update, roll back, and uninstall the Web Server Plug-ins.

Before you begin

Note: The Web Server Plug-ins for WebSphere Application Server Version 8.0 are now installed by Installation Manager rather than by the programs based on InstallShield MultiPlatform (ISMP) that are used to install, update, and uninstall previous versions.

Restrictions:

- Solaris The Installation Manager GUI is not supported on Solaris 10 x64 systems. Perform the following actions to install or uninstall the product on these systems:
  - Use the Installation Manager GUI on a supported system to record a response file that will allow you to install or uninstall the product silently.
  - Edit the recorded response file if necessary.
  - Use the response file to install or uninstall the product silently on your system.

- Linux For any Linux system that is enabled for Security Enhanced Linux (SELinux), such as Red Hat Enterprise Linux Version 5 or SUSE Linux Enterprise Server Version 11, you must identify the Java shared libraries in the Installation Manager Version 1.4.3 or earlier installation image to the system. Also, you must identify the Java shared libraries in the Installation Manager Version 1.4.3 or earlier installation after it has been installed. For example:

```
chcon -R -t texrel_shlib_t ${IM_Image}/jre_5.0.3.sr8a_20080811b/jre/bin
chcon -R -t texrel_shlib_t ${IM_Install_root}/eclipse/jre_5.0.3.sr8a_20080811b/jre/bin
```

- Installation Manager console mode, which is included in Installation Manager Version 1.4.3 and later, does not work with WebSphere Application Server Version 8.0 offerings.

About this task

Perform one of these procedures to install, update, roll back, or uninstall the Web Server Plug-ins using Installation Manager.

Procedure

- "Installing the Web Server Plug-ins using the GUI" on page 70
- "Installing the Web Server Plug-ins silently" on page 74
- "Installing fix packs on the Web Server Plug-ins using the Installation Manager GUI" on page 84
- "Uninstalling fix packs from the Web Server Plug-ins using the Installation Manager GUI" on page 85
- "Uninstalling Web Server Plug-ins using the GUI" on page 86
- "Uninstalling the Web Server Plug-ins silently" on page 86

Results

Notes on logging and tracing:

- An easy way to view the logs is to open Installation Manager and go to **File > View Log**. An individual log file can be opened by selecting it in the table and then clicking the **Open log file** icon.
- Logs are located in the `logs` directory of Installation Manager's application data location. For example:
  - Windows **Administrative installation:**

```
C:\Documents and Settings\All Users\Application Data\IBM\Installation Manager\logs
```

  - Windows **Non-administrative installation:**

```
C:\Documents and Settings\user_name\Application Data\IBM\Installation Manager\logs
```

  - AIX HP-UX Linux Solaris **Administrative installation:**

```
/var/ibm/InstallationManager/logs
```

  - AIX HP-UX Linux Solaris **Non-administrative installation:**

```
user_home/var/ibm/InstallationManager/logs
```

- The main log files are time-stamped XML files in the `logs` directory, and they can be viewed using any standard web browser.
- The `log.properties` file in the `logs` directory specifies the level of logging or tracing that Installation Manager uses.

Notes on troubleshooting:

- HP-UX By default, some HP-UX systems are configured to not use DNS to resolve host names. This could result in Installation Manager not being able to connect to an external repository.

  You can ping the repository, but nslookup does not return anything.

Work with your system administrator to configure your machine to use DNS, or use the IP address of the repository.

- Installation Manager might display a warning message during the uninstallation process.

  Uninstalling the Web Server Plug-ins using Installation Manager requires that the data repositories remain valid and available.
- For more information on using Installation Manager, read the IBM Installation Manager Information Center.

  Read the release notes to learn more about the latest version of Installation Manager. To access the release notes, complete the following task:
  - Windows Click **Start > Programs > IBM Installation Manager > Release Notes**.
  - AIX HP-UX Linux Solaris Go to the documentation subdirectory in the directory where Installation Manager is installed, and open the `readme.html` file.

Installing the Web Server Plug-ins using the GUI

You can use the Installation Manager GUI to install the Web Server Plug-ins.

Before you begin

Install Installation Manager:

1. Perform one of the following procedures:
   - If you want to use the Installation Manager that is included with this product, perform the following actions:
     a. Obtain the necessary files from the physical media or the web.

        There are three basic options for obtaining and installing the product.
        - **Access the physical media, and use local installation**

          You can access the product repositories on the product media.
          1) Install Installation Manager on your system.

             You can install Installation Manager using the product media, using a file obtained from the Passport Advantage® site, or using a file containing the most current version of Installation Manager from the IBM Installation Manager website.
          2) Use Installation Manager to install the product from the product repositories on the media.
        - **Download the files from the Passport Advantage site, and use local installation**

          Licensed customers with a Passport Advantage ID and password can download the necessary product repositories from the Passport Advantage site.
          1) Download the files from the Passport Advantage site.
          2) Install Installation Manager on your system.

             You can install Installation Manager using the product media, using a file obtained from the Passport Advantage site, or using a file containing the most current version of Installation Manager from the IBM Installation Manager website.
          3) Use Installation Manager to install the product from the downloaded repositories.
        - **Access the live repositories, and use web-based installation**

          If you have a Passport Advantage ID and password, you can install the product from the web-based repositories.
          1) Install Installation Manager on your system.

You can install Installation Manager using the product media, using a file obtained from the Passport Advantage site, or using a file containing the most current version of Installation Manager from the IBM Installation Manager website.

2) Use Installation Manager to install the product from the web-based repository located at

`http://www.ibm.com/software/repositorymanager/com.ibm.websphere.PLG.v80`

Whenever possible, you should use the remote web-based repositories so that you are accessing the most up-to-date installation files.

Notes:

- If you do not have a Passport Advantage ID and password, you must install the product from the product repositories on the media or local repositories.
- With the Packaging Utility, you can create and manage packages for installation repositories. You can copy multiple packages into one repository or copy multiple disks for one product into a repository. You can copy packages from Passport Advantage into a repository for example. For more information on the Packaging Utility, go to the IBM Installation Manager Information Center.

b. Change to the location containing the Installation Manager installation files, and run one of the following commands:

Administrative installation:

- Windows `install.exe`
- AIX HP-UX Linux Solaris `./install`

Non-administrative installation:

- Windows `userinst.exe`
- AIX HP-UX Linux Solaris `./userinst`

Group-mode installation:

AIX HP-UX Linux Solaris `./groupinst`

Notes on group mode:

- Group mode allows multiple users to use a single instance of IBM Installation Manager to manage software packages.
- Windows Group mode is not available on Windows operating systems.
- If you do not install Installation Manager using group mode, you will not be able to use group mode to manage any of the products that you install later using this Installation Manager.
- Make sure that you change the installation location from the default location in the current user's home directory to a location that is accessible by all users in the group.
- Set up your groups, permissions, and environment variables as described in the Group mode road maps in the IBM Installation Manager Information Center before installing in group mode.
- For more information on using group mode, read the Group mode road maps in the IBM Installation Manager Information Center.

The installer opens an **Install Packages** window.

c. Make sure that the Installation Manager package is selected, and click **Next**.

d. Accept the terms in the license agreements, and click **Next**.

The program creates the directory for your installation.

e. Click **Next**.

f. Review the summary information, and click **Install**.
   - If the installation is successful, the program displays a message indicating that installation is successful.
   - If the installation is not successful, click **View Log File** to troubleshoot the problem.

- If you already have a version of Installation Manager installed on your system and you want to use it to install and maintain the product, obtain the necessary product files from the physical media or the web.

  There are three basic options for installing the product.

  – **Access the physical media, and use local installation**

  You can access the product repositories on the product media. Use Installation Manager to install the product from the product repositories on the media.

  – **Download the files from the Passport Advantage site, and use local installation**

  Licensed customers with a Passport Advantage ID and password can download the necessary product repositories from the Passport Advantage site.

  a. Download the product repositories from the Passport Advantage site.

  b. Use Installation Manager to install the product from the downloaded repositories.

  – **Access the live repositories, and use web-based installation**

  If you have a Passport Advantage ID and password, you can use Installation Manager to install the product from the web-based repositories. Use Installation Manager to install the product from the web-based repository located at

```
http://www.ibm.com/software/repositorymanager/com.ibm.websphere.PLG.v80
```

  Whenever possible, you should use the remote web-based repositories so that you are accessing the most up-to-date installation files.

  Notes:

  – If you do not have a Passport Advantage ID and password, you must install the product from the product repositories on the media or local repositories.

  – With the Packaging Utility, you can create and manage packages for installation repositories. You can copy multiple packages into one repository or copy multiple disks for one product into a repository. You can copy packages from Passport Advantage into a repository for example. For more information on the Packaging Utility, go to the IBM Installation Manager Information Center.

2. Add the product repository to your Installation Manager preferences.
   a. Start Installation Manager.
   b. In the top menu, click **File > Preferences**.
   c. Select **Repositories**.
   d. Perform the following actions:
      1) Click **Add Repository**.
      2) Enter the path to the `repository.config` file in the location containing the repository files.

         For example:
         - Windows `C:\repositories\`*`product_name`*`\local-repositories`
         - AIX HP-UX Linux Solaris `/var/repositories/`*`product_name`*`/local-repositories`

         or

```
http://www.ibm.com/software/repositorymanager/com.ibm.websphere.PLG.v80
```

      3) Click **OK**.

e. Deselect any locations listed in the Repositories window that you will not be using.
f. Click **Apply**.
g. Click **OK**.
h. Click **File > Exit** to close Installation Manager.

About this task

Perform this procedure to use the Installation Manager GUI to install the Web Server Plug-ins.

Procedure

1. Start Installation Manager.

   Tip: AIX HP-UX Linux Solaris You can start Installation Manager in group mode with the ./IBMIM command.
   - Group mode allows users to share packages in a common location and manage them with the same instance of Installation Manager.
   - For more information on using group mode, read the Group mode road maps in the IBM Installation Manager Information Center.
2. Click **Install**.

   Note: If you are prompted to authenticate, use the IBM ID and password that you registered with on the program website.

   Installation Manager searches its defined repositories for available packages.
3. Perform the following actions.
   a. Select **Web Server Plug-ins for IBM WebSphere Application Server** and the appropriate version.

      Note: If you are installing the ILAN version of this product, select **Web Server Plug-ins for IBM WebSphere Application Server (ILAN)**.

      If you already have the Web Server Plug-ins installed on your system, a message displays indicating that the Web Server Plug-ins are already installed. To create another installation of the Web Server Plug-ins in another location, click **Continue**.

      Tip: If the **Search service repositories during installation and updates** option is selected on the Installation Manager Repository preference page and you are connected to the Internet, you can click **Check for Other Versions and Extensions** to search for updates in the default update repositories for the selected packages. In this case, you do not need to add the specific service-repository URL to the Installation Manager Repository preference page.

   b. Select the fixes to install.

      Any recommended fixes are selected by default.

      If there are recommended fixes, you can select the option to show only recommended fixes and hide non-recommended fixes.

   c. Click **Next**.

   Note: If you try to install a newer level of the Web Server Plug-ins with a previous version of Installation Manager, Installation Manager might prompt you to update to the latest level of Installation Manager when it connects to the repository. Update to the newer version before you continue if you are prompted to do so. Read Installing updates in the Installation Manager information center for information about automatic updates.
4. Accept the terms in the license agreements, and click **Next**.
5. Specify the installation root directory for the product binaries, which are also referred to as the core product files or system files.

The panel also displays the shared resources directory and disk-space information.

Restrictions:

- Deleting the default target location and leaving an installation-directory field empty prevents you from continuing.
- Do not use symbolic links as the destination directory.

  Symbolic links are not supported.
- Do not use a semicolon in the directory name.

  The Web Server Plug-ins cannot install properly if the target directory includes a semicolon.

  Windows A semicolon is the character used to construct the class path on Windows systems.
- Windows The maximum path length on the Windows Server 2008, Windows Vista, and Windows 7 operating systems is 60 characters.

6. Click **Next**.
7. If you are installing on a 64-bit system, choose between a 32-bit and 64-bit IBM Runtime Environment for Java and click **Next**.

   Notes:
   - This option displays only if you are installing on a 64-bit system.
   - You must select one of the two options.
   - You cannot modify this installation later and change this selection.
8. Review the summary information, and click **Install**.
   - If the installation is successful, the program displays a message indicating that installation is successful.

     Note: The program might also display important post-installation instructions as well.
   - If the installation is not successful, click **View Log File** to troubleshoot the problem.
9. Click **Finish**.
10. Click **File > Exit** to close Installation Manager.

Installing the Web Server Plug-ins silently

You can use Installation Manager to install the Web Server Plug-ins silently.

Before you begin

Install Installation Manager on each of the systems onto which you want to install the product.

1. Perform one of the following procedures:
   - If you want to use the Installation Manager that is included with this product, perform the following actions:
     a. Obtain the necessary files from the physical media or the web.

        There are three basic options for obtaining and installing the product.
        - **Access the physical media, and use local installation**

          You can access the product repositories on the product media.

          1) Install Installation Manager on your system.

             You can install Installation Manager using the product media, using a file obtained from the Passport Advantage site, or using a file containing the most current version of Installation Manager from the IBM Installation Manager website.

          2) Use Installation Manager to install the product from the product repositories on the media.

- **Download the files from the Passport Advantage site, and use local installation**

  Licensed customers with a Passport Advantage ID and password can download the necessary product repositories from the Passport Advantage site.

  1) Download the files from the Passport Advantage site.
  2) Install Installation Manager on your system.

     You can install Installation Manager using the product media, using a file obtained from the Passport Advantage site, or using a file containing the most current version of Installation Manager from the IBM Installation Manager website.
  3) Use Installation Manager to install the product from the downloaded repositories.
- **Access the live repositories, and use web-based installation**

  If you have a Passport Advantage ID and password, you can install the product from the web-based repositories.

  1) Install Installation Manager on your system.

     You can install Installation Manager using the product media, using a file obtained from the Passport Advantage site, or using a file containing the most current version of Installation Manager from the IBM Installation Manager website.
  2) Use Installation Manager to install the product from the web-based repository located at

`http://www.ibm.com/software/repositorymanager/com.ibm.websphere.PLG.v80`

Whenever possible, you should use the remote web-based repositories so that you are accessing the most up-to-date installation files.

Notes:

- If you do not have a Passport Advantage ID and password, you must install the product from the product repositories on the media or local repositories.
- With the Packaging Utility, you can create and manage packages for installation repositories. You can copy multiple packages into one repository or copy multiple disks for one product into a repository. You can copy packages from Passport Advantage into a repository for example. For more information on the Packaging Utility, go to the IBM Installation Manager Information Center.

b. Change to the location containing the Installation Manager installation files, and run one of the following commands to install Installation Manager silently:

Administrative installation:

- Windows `installc.exe -acceptLicense -log` *`log_file_path_and_name`*
- AIX HP-UX Linux Solaris `./installc -acceptLicense -log` *`log_file_path_and_name`*

Non-administrative installation:

- Windows `userinstc.exe -acceptLicense -log` *`log_file_path_and_name`*
- AIX HP-UX Linux Solaris `./userinstc -acceptLicense -log` *`log_file_path_and_name`*

Group-mode installation:

AIX HP-UX Linux Solaris `./groupinstc -acceptLicense -dataLocation` *`application_data_location`* `-log` *`log_file_path_and_name`*

Notes on group mode:

- Group mode allows multiple users to use a single instance of IBM Installation Manager to manage software packages.
- Windows Group mode is not available on Windows operating systems.

- If you do not install Installation Manager using group mode, you will not be able to use group mode to manage any of the products that you install later using this Installation Manager.
- Make sure that you change the installation location from the default location in the current user's home directory to a location that is accessible by all users in the group.
- Set up your groups, permissions, and environment variables as described in the Group mode road maps in the IBM Installation Manager Information Center before installing in group mode.
- For more information on using group mode, read the Group mode road maps in the IBM Installation Manager Information Center.

- If you already have a version of Installation Manager installed on your system and you want to use it to install and maintain the product, obtain the necessary product files from the physical media or the web.

  There are three basic options for installing the product.

  - **Access the physical media, and use local installation**

    You can access the product repositories on the product media. Use Installation Manager to install the product from the product repositories on the media.

  - **Download the files from the Passport Advantage site, and use local installation**

    Licensed customers with a Passport Advantage ID and password can download the necessary product repositories from the Passport Advantage site.

    a. Download the product repositories from the Passport Advantage site.
    b. Use Installation Manager to install the product from the downloaded repositories.

  - **Access the live repositories, and use web-based installation**

    If you have a Passport Advantage ID and password, you can use Installation Manager to install the product from the web-based repositories. Use Installation Manager to install the product from the web-based repository located at

```
http://www.ibm.com/software/repositorymanager/com.ibm.websphere.PLG.v80
```

    Whenever possible, you should use the remote web-based repositories so that you are accessing the most up-to-date installation files.

  Notes:

  - If you do not have a Passport Advantage ID and password, you must install the product from the product repositories on the media or local repositories.
  - With the Packaging Utility, you can create and manage packages for installation repositories. You can copy multiple packages into one repository or copy multiple disks for one product into a repository. You can copy packages from Passport Advantage into a repository for example. For more information on the Packaging Utility, go to the IBM Installation Manager Information Center.

2. Add the product repository to your Installation Manager preferences.
   a. Start Installation Manager.
   b. In the top menu, click **File > Preferences**.
   c. Select **Repositories**.
   d. Perform the following actions:
      1) Click **Add Repository**.
      2) Enter the path to the `repository.config` file in the location containing the repository files.

         For example:

- Windows C:\repositories\product_name\local-repositories
- AIX HP-UX Linux Solaris /var/repositories/*product_name*/local-repositories

or

```
http://www.ibm.com/software/repositorymanager/com.ibm.websphere.PLG.v80
```

3) Click **OK**.

e. Deselect any locations listed in the Repositories window that you will not be using.

f. Click **Apply**.

g. Click **OK**.

h. Click **File > Exit** to close Installation Manager.

About this task

Using Installation Manager, you can work with response files to install the Web Server Plug-ins silently in a variety of ways. You can record a response file using the GUI as described in the following procedure, or you can generate a new response file by hand or by taking an example and modifying it.

Procedure

1. Optional: **Record a response file to install the Web Server Plug-ins:** On one of your systems, perform the following actions to record a response file that will install the Web Server Plug-ins.

   a. From a command line, change to the eclipse subdirectory in the directory where you installed Installation Manager.

   b. Start Installation Manager from the command line using the -record option.

      For example:

      - Windows **Administrator or non-administrator:**

```
IBMIM.exe -skipInstall "C:\temp\imRegistry"
  -record C:\temp\install_response_file.xml
```

      - AIX HP-UX Linux Solaris **Administrator:**

```
./IBMIM -skipInstall /var/temp/imRegistry
  -record /var/temp/install_response_file.xml
```

      - AIX HP-UX Linux Solaris **Non-administrator:**

```
./IBMIM -skipInstall user_home/var/temp/imRegistry
  -record user_home/var/temp/install_response_file.xml
```

      Tip: When you record a new response file, you can specify the -skipInstall parameter. Using this parameter has the following benefits:

      - No files are actually installed, and this speeds up the recording.
      - If you use a temporary data location with the -skipInstall parameter, Installation Manager writes the installation registry to the specified data location while recording. When you start Installation Manager again without the -skipInstall parameter, you then can use your response file to install against the real installation registry.

        The -skipInstall operation should not be used on the actual agent data location used by Installation Manager. This is unsupported. Use a clean writable location, and re-use that location for future recording sessions.

      For more information, read the IBM Installation Manager Information Center.

   c. Add the appropriate repositories to your Installation Manager preferences.

      1) In the top menu, click **File > Preferences**.

      2) Select **Repositories**.

      3) Perform the following actions for each repository:

         a) Click **Add Repository**.

b) Enter the path to the `repository.config` file in the remote web-based repository or the local directory into which you unpacked the repository files.

For example:

- Remote repositories:

```
https://downloads.mycorp.com:8080/WAS_80_repository
```

or

```
http://www.ibm.com/software/repositorymanager/com.ibm.websphere.PLG.v80
```

- Local repositories:
  - Windows `C:\repositories\plugins\local-repositories`
  - AIX HP-UX Linux Solaris `/var/repositories/plugins/local-repositories`

c) Click **OK**.

4) Click **Apply**.

5) Click **OK**.

d. Click **Install**.

Note: If you are prompted to authenticate, use the IBM ID and password that you registered with on the program website.

Installation Manager searches its defined repositories for available packages.

e. Perform the following actions.

1) Select **Web Server Plug-ins for IBM WebSphere Application Server** and the appropriate version.

Note: If you are installing the ILAN version of this product, select **Web Server Plug-ins for IBM WebSphere Application Server (ILAN)**.

If you already have the Web Server Plug-ins installed on your system, a message displays indicating that the Web Server Plug-ins are already installed. To create another installation of the Web Server Plug-ins in another location, click **Continue**.

2) Click **Next**.

f. Accept the terms in the license agreements, and click **Next**.

g. Specify the installation root directory for the Web Server Plug-ins binaries, which are also referred to as the core product files or system files.

The panel also displays the shared resources directory and disk-space information.

Restrictions:

- Deleting the default target location and leaving an installation-directory field empty prevents you from continuing.
- Do not use symbolic links as the destination directory.

  Symbolic links are not supported.
- Do not use a semicolon in the directory name.

  The Web Server Plug-ins cannot install properly if the target directory includes a semicolon.

  Windows A semicolon is the character used to construct the class path on Windows systems.
- Windows The maximum path length on the Windows Server 2008, Windows Vista, and Windows 7 operating systems is 60 characters.

h. Click **Next**.

i. If you are installing on a 64-bit system, choose between a 32-bit and 64-bit IBM Runtime Environment for Java and click **Next**.

Notes:

- This option displays only if you are installing on a 64-bit system.
- You must select one of the two options.
- You cannot modify this installation later and change this selection.

j. Review the summary information, and click **Install**.

- If the installation is successful, the program displays a message indicating that installation is successful.

  Note: The program might also display important post-installation instructions as well.
- If the installation is not successful, click **View Log File** to troubleshoot the problem.

k. Click **Finish**.

l. Click **File > Exit** to close Installation Manager.

m. Optional: If you are using an authenticated remote repository, create a keyring file for silent installation.

1) From a command line, change to the eclipse subdirectory in the directory where you installed Installation Manager.

2) Start Installation Manager from the command line using the -record option.

For example:

- Windows **Administrator or non-administrator:**

```
IBMIM.exe -skipInstall "C:\temp\imRegistry"
  -keyring C:\IM\im.keyring
  -record C:\temp\keyring_response_file.xml
```

- AIX HP-UX Linux Solaris **Administrator:**

```
./IBMIM -skipInstall /var/temp/imRegistry
  -keyring /var/IM/im.keyring
  -record /var/temp/keyring_response_file.xml
```

- AIX HP-UX Linux Solaris **Non-administrator:**

```
./IBMIM -skipInstall user_home/var/temp/imRegistry
  -keyring user_home/var/IM/im.keyring
  -record user_home/var/temp/keyring_response_file.xml
```

3) When a window opens that requests your credentials for the authenticated remote repository, enter the correct credentials and **save** them.

4) Click **File > Exit** to close Installation Manager.

For more information, read the IBM Installation Manager Information Center.

2. **Use the response files to install the Web Server Plug-ins silently:**

a. Optional: **Use the response file to install the keyring silently:** Go to a command line on each of the systems on which you want to install the product, change to the `eclipse/tools` subdirectory in the directory where you installed Installation Manager, and install the keyring silently.

For example:

- Windows **Administrator or non-administrator:**

```
imcl.exe -acceptLicense
  input C:\temp\keyring_response_file.xml
  -log C:\temp\keyring_log.xml
```

- AIX HP-UX Linux Solaris **Administrator:**

```
./imcl -acceptLicense
  input /var/temp/keyring_response_file.xml
  -log /var/temp/keyring_log.xml
```

- AIX HP-UX Linux Solaris **Non-administrator:**

```
./imcl -acceptLicense
  input user_home/var/temp/keyring_response_file.xml
  -log user_home/var/temp/keyring_log.xml
```

b. **Use the response file to install the product silently:** Go to a command line on each of the systems on which you want to install the product, change to the `eclipse/tools` subdirectory in the directory where you installed Installation Manager, and install the product silently.

For example:

- Windows **Administrator or non-administrator:**

```
imcl.exe -acceptLicense
  input C:\temp\install_response_file.xml
  -log C:\temp\install_log.xml
  -keyring C:\IM\im.keyring
```

- AIX HP-UX Linux Solaris **Administrator:**

```
./imcl -acceptLicense
  input /var/temp/install_response_file.xml
  -log /var/temp/install_log.xml
  -keyring /var/IM/im.keyring
```

- AIX HP-UX Linux Solaris **Non-administrator:**

```
./imcl -acceptLicense
  input user_home/var/temp/install_response_file.xml
  -log user_home/var/temp/install_log.xml
  -keyring user_home/var/IM/im.keyring
```

Notes:

- The relevant terms and conditions, notices, and other information are provided in the license-agreement files in the `lafiles` or *product_name*/`lafiles` subdirectory of the installation image or repository for this product.
- The program might write important post-installation instructions to standard output.

Read the IBM Installation Manager Information Center for more information.

Example

Windows The following is an example of a response file for silently installing the Web Server Plug-ins.

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- ##### Copyright ######################################################
# Licensed Materials - Property of IBM (c) Copyright IBM Corp. 2011.
# All Rights Reserved. US Government Users Restricted Rights-Use, duplication
# or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
##################################################################### -->

<!-- ##### Frequently Asked Questions ####################################
# The latest information about using Installation Manager is
# located in the online Information Center. There you can find
# information about the commands and attributes used in
# silent installation response files.
#
#     Installation Manager Information Center can be found at:
#     http://publib.boulder.ibm.com/infocenter/install/v1r4/index.jsp
#
# Question 1. How do I record a response file using Installation Manager?
# Answer 1. Start Installation Manager from the command line under the
# eclipse subdirectory with the record parameter and it will generate a
# response file containing actions it performed, repositories it used, and
# its preferences settings. Optionally use the -skipInstall parameter if
# you do not want the product to be installed to the machine. Specify a
# new agentDataLocation location value when doing a new installation. Do
# not use an existing agentDataLocation for an installation because it might
# damage the installation data and prevent you from modifying, updating,
# rolling back, or uninstalling the installed packages.
#
# Windows: IBMIM -record <responseFile> -skipInstall <agentDataLocation>
# Linux or UNIX: ./IBMIM -record <responseFile> -skipInstall <agentDataLocation>
#
# For example:
#   Windows = IBMIM.exe -record c:\temp\responsefiles\WASv8.install.Win32.xml
#     -skipInstall c:\temp\skipInstall\WebSphere_Temp_Registry
#   Linux or UNIX = ./IBMIM -record /home/user/responsefiles/WASv8.install.RHEL64.xml
#     -skipInstall c:\temp\skipInstall\WebSphere_Temp_Registry
#
# Question 2. How do I run Installation Manager silently using response file?
# Answer 2. Create a silent installation response file and run the following command
# from the eclipse\tools subdirectory in the directory where you installed
# Installation Manager:
#
#   Windows = imcl.exe -acceptLicense -showProgress
#     input <response_file_path_and_name> -log <log_file_path_and_name>
#   Linux, UNIX, IBM i and z/OS = ./imcl -acceptLicense -showProgress
#     input <response_file_path_and_name> -log <log_file_path_and_name>
#
# For example:
#   Windows = imcl.exe -acceptLicense -showProgress
#     input c:\temp\responsefile\WASv8.install.Win32.xml
#   Linux, UNIX, IBM i and z/OS = ./imcl -acceptLicense -showProgress
#     input /home/user/responsefile/WASv8.install.RHEL64.xml
#
# The -acceptLicense command must be included to indicate acceptance of all
#     license agreements of all offerings being installed, updated or modified.
# The -showProgress command shows progress when running in silent mode.
# Additional commands can be displayed by requesting help:  IBMIM -help
#
# Question 3. How do I store and pass credentials to repositories that
# require authentication?
# Answer 3. Installation Manager uses a key ring file to store encrypted
# credentials for authenticating with repositories. Follow this two-step
# process for creating and using a key ring file with Installation Manager.
#
# First, create a key ring file with your credentials by starting
# Installation Manager from the command line under eclipse subdirectory
# with the keyring parameter.
# Use the optional password parameter to password protect your file.
#
#   Windows = IBMIM.exe -keyring <path and file name> -password <password>
#   Linux, UNIX, IBM i and z/OS = ./IBMIM -keyring <path and file name>
#                                -password <password>
#
# Installation Manager will start in graphical mode. Verify that the
# repositories to which you need to authenticate are included in the
# preferences, File / Preferences / Repositories. If they are not
# listed, then click Add Repositories to add the URL or UNC path.
# Installation Manager will prompt for your credentials. If the repository
# is already in the list, then any attempt to access the repository location,
# such as clicking the Test Connections button, will also prompt for your
# credentials. Enter the correct credential and check the Save password
# checkbox. The credentials are saved to the key ring file you specified.
#
# Second, when you start a silent installation, run imcl under eclipse/tools
# subdirectory, and provide Installation Manager with the location of the key
# ring file and the password if the file is protected. For example:
#
#   Windows = imcl.exe -acceptLicense -showProgress
#     input <path and file name of response file>
#     -keyring <path and name of key ring file> -password <password>
#   Linux, UNIX, IBM i and z/OS = ./imcl -acceptLicense -showProgress
#     input <path and file name of response file>
#     -keyring <path and name of key ring file> -password <password>
#
###################################################################### -->

<!-- ##### Agent Input ############################################
#
# Note that the "acceptLicense" attribute has been deprecated.
# Use "-acceptLicense" command line option to accept license agreements.
#
# The clean and temporary attributes specify the repositories and other
# preferences Installation Manager uses and whether those settings
# should persist after the installation finishes.
#
# Valid values for clean:
#      true = only use the repositories and other preferences that are
#           specified in the response file.
#      false = use the repositories and other preferences that are
#           specified in the response file and Installation Manager.
#
# Valid values for temporary:
#      true = repositories and other preferences specified in the
#           response file do not persist in Installation Manager.
#      false = repositories and other preferences specified in the
#           response file persist in Installation Manager.
#
###################################################################### -->

<agent-input clean="true" temporary="true">

<!-- ##### Repositories ################################################
# Repositories are locations that Installation Manager queries for
# installable packages. Repositories can be local (on the machine
# with Installation Manager) or remote (on a corporate intranet or
# hosted elsewhere on the internet).
#
# If the machine using this response file has access to the internet,
# then include the IBM WebSphere Live Update Repositories in the list
# of repository locations.
#
# If the machine using this response file cannot access the internet,
# then comment out the IBM WebSphere Live Update Repositories and
# specify the URL or UNC path to custom intranet repositories and
# directory paths to local repositories to use.
#
######################################################################## -->

<server>
    <!-- ##### IBM WebSphere Live Update Repositories ###################
     # These repositories contain Web Server Plug-ins for IBM WebSphere
     # Application Server offerings, and updates for those offerings
     #
     # To use the secure repository (https), you must have an IBM ID,
     # which can be obtained by registering at: http://www.ibm.com/account
     # or your Passport Advantage account.
     #
     # And, you must use a key ring file with your response file.
     ################################################################## -->
<repository location="http://www.ibm.com/software/repositorymanager/com.ibm.websphere.PLG.v80" />
    <!-- <repository location="https://www.ibm.com/software/rational/repositorymanager/repositories/websphere" /> -->

    <!-- ##### Custom Repositories ######################################
     # Uncomment and update the repository location key below
     # to specify URLs or UNC paths to any intranet repositories
     # and directory paths to local repositories to use.
     ################################################################## -->
    <!-- <repository location='https:\\w3.mycompany.com\repositories\'/> -->
    <!-- <repository location='/home/user/repositories/websphere/'/> -->

    <!-- ##### Local Repositories #######################################
     # Uncomment and update the following line when using a local
     # repository located on your own machine to install a
     # Web Server Plug-ins offering.
     ############################################################## -->
    <!-- <repository location='insert the full directory path inside single quotes'/> -->
</server>

<!-- ##### Install Packages ########################################
#
# Install Command
#
# Use the install command to inform Installation Manager of the
# installation packages to install.
#
# The modify attribute is optional and can be paired with an install
# command to add features or paired with an uninstall command to
# remove commands. If omitted, the default value is set to false.
#    false = indicates not to modify an existing install by adding
#            or removing features.
#    true = indicates to modify an existing install by adding or
#           removing features.
#
# The offering ID attribute is required because it specifies the
# offering to be installed. The offering listed must be present in
# at least one of the repositories listed earlier. The example
# command below contains the offering ID for the Web Server Plug-ins.
#
# The version attribute is optional. If a version number is provided,
# then the offering will be installed at the version level specified
# as long as it is available in the repositories. If the version
# attribute is not provided, then the default behavior is to install
# the latest version available in the repositories. The version number
# can be found in the repository.xml file in the repositories.
# For example, <offering ... version='8.0.0.20110617_2222'>.
#
# The profile attribute is required and typically is unique to the
# offering. If modifying or updating an existing installation, the
# profile attribute must match the profile ID of the targeted installation
# of Web Server Plug-ins.
#
# The features attribute is optional. Offerings always have at least
# one feature; a required core feature which is installed regardless
# of whether it is explicitly specified. If other feature names
# are provided, then only those features will be installed.
# Features must be comma delimited without spaces.
#
# The feature values for Web Server Plug-ins include:
#  com.ibm.jre.6_32bit,com.ibm.jre.6_64bit
#
# On 32-bit machines, the 32-bit jre feature will be install
# automatically even if it is not specified in the response file.
#
# On 64-bit machines, one and only one of the Java Runtime Environment
# (JRE) features must be specified.
#
# The installFixes attribute indicates whether fixes available in
# repositories are installed with the product. By default, all
# available fixes will be installed with the offering.
#
# Valid values for installFixes:
#      none = do not install available fixes with the offering.
#      recommended = installs all available recommended fixes with the offering.
#      all = installs all available fixes with the offering.
```

```
#
# Interim fixes for offerings also can be installed while they
# are being installed by including the offering ID for the interim
# fix and specifying the profile ID. A commented out example is
# provided in the install command below.
#
# Installation Manager supports installing multiple offerings at once.
# Additional offerings can be included in the install command,
# with each offering requiring its own offering ID, version, profile value,
# and feature values.
#
# Profile Command
#
# A separate profile command must be included for each offering listed
# in the install command. The profile command informs Installation
# Manager about offering specific properties or configuration values.
#
# The installLocation specifies where the offering will be installed.
# If the response file is used to modify or update an existing
# installation, then ensure the installLocation points to the
# location where the offering was installed previously.
#
# The eclipseLocation data key should use the same directory path to
# Web Server Plug-ins as the installationLocation attribute.
#
# Include data keys for product specific profile properties.
#
###################################################################### -->

<install modify='false'>
<offering id='com.ibm.websphere.PLG.v80'
 profile='Web Server Plug-ins for IBM WebSphere Application Server V8.0'
 features='core.feature,com.ibm.jre.6_32bit' installFixes='none'/>
<!-- <offering id='PM12345_WAS80' profile='Web Server Plug-ins for IBM WebSphere Application Server V8.0'/> -->
</install>

<profile id='Web Server Plug-ins for IBM WebSphere Application Server V8.0'
 installLocation='C:\Program Files\IBM\WebSphere\Plugins'>
<data key='eclipseLocation' value='C:\Program Files\IBM\WebSphere\Plugins'/>
<data key='user.import.profile' value='false'/>
<data key='cic.selector.nl' value='en'/>
</profile>

<!-- ##### Shared Data Location ##########################################
# Uncomment the preference for eclipseCache to set the shared data
# location the first time you use Installation Manager to do an
# installation.
#
# Eclipse cache location can be obtained from the installed.xml file found in
# Linux/Unix: /var/ibm/InstallationManager
# Windows: C:\Documents and Settings\All Users\Application Data\IBM\Installation Manager
# from the following property:
# <property name='cacheLocation' value='C:\Program Files\IBM\IMShared'/>
#
# Open the installed.xml file in a text editor because the style sheet
# might hide this value if opened in a web browser.
# For further information on how to edit preferences, refer to the public library at:
# http://publib.boulder.ibm.com/infocenter/install/v1r4/index.jsp?topic=/com.ibm.silentinstall12.doc/topics/r_silent_prefs.html
#
# After the shared data location is set, it cannot be changed
# using a response file or the graphical wizard.
#
# Ensure that the shared data location is a location that can be written
# to by all user accounts that are expected to use Installation Manager.
#
# By default, Installation Manager saves downloaded artifacts to
# the shared data location. This serves two purposes.
#
# First, if the same product is installed a more than once to the machine,
# then the files in the shared data location will be used rather than
# downloading them again.
#
# Second, during the rollback process, the saved artifacts are used.
# Otherwise, if the artifacts are not saved or are removed, then
# Installation Manager must have to access the repositories used to
# install the previous versions.
#
# Valid values for preserveDownloadedArtifacts:
#     true = store downloaded artifacts in the shared data location
#     false = remove downloaded artifacts from the shared data location
#
###################################################################### -->

<!--
<preference name='com.ibm.cic.common.core.preferences.eclipseCache' value='C:\Program Files\IBM\IMShared'/>
<preference name='com.ibm.cic.common.core.preferences.preserveDownloadedArtifacts' value='true'/>
-->


<!-- ##### Preferences Settings ##########################################
```

```
# Additional preferences for Installation Manager can be specified.
# These preference correspond to those that are located in the graphical
# interface under File / Preferences.
#
# If a preference command is omitted from or commented out of the response
# file, then Installation Manager uses the preference value that was
# previously set or the default value for the preference.
#
# Preference settings might be added or deprecated in new versions of
# Installation Manager. Consult the online Installation Manager
# Information Center for the latest set of preferences and
# descriptions about how to use them.
#
# http://publib.boulder.ibm.com/infocenter/install/v1r4/index.jsp
#
################################################################## -->

<!--
<preference name='com.ibm.cic.common.core.preferences.connectTimeout' value='30'/>
<preference name='com.ibm.cic.common.core.preferences.readTimeout' value='45'/>
<preference name='com.ibm.cic.common.core.preferences.downloadAutoRetryCount' value='0'/>
<preference name='offering.service.repositories.areUsed' value='true'/>
<preference name='com.ibm.cic.common.core.preferences.ssl.nonsecureMode' value='false'/>
<preference name='com.ibm.cic.common.core.preferences.http.disablePreemptiveAuthentication' value='false'/>
<preference name='http.ntlm.auth.kind' value='NTLM'/>
<preference name='http.ntlm.auth.enableIntegrated.win32' value='true'/>
<preference name='com.ibm.cic.common.core.preferences.keepFetchedFiles' value='false'/>
<preference name='PassportAdvantageIsEnabled' value='false'/>
<preference name='com.ibm.cic.common.core.preferences.searchForUpdates' value='false'/>
<preference name='com.ibm.cic.agent.ui.displayInternalVersion' value='false'/>
 -->

</agent-input>
```

Important: AIX Linux Windows If you are installing on a 64-bit system, you must include one of the options for an IBM Runtime Environment for Java.

- If you want to use the 32-bit IBM Runtime Environment for Java, include `com.ibm.jre.6_32bit` as a feature in the response file.

  For example:

```
<offering profile='Web Server Plug-ins for IBM WebSphere Application Server V8.0'
  features='core.feature,com.ibm.jre.6_32bit' id='com.ibm.websphere.PLG.v80'/>
```

- If you want to use the 64-bit IBM Runtime Environment for Java, include `com.ibm.jre.6_64bit` as a feature in the response file.

  For example:

```
<offering profile='Web Server Plug-ins for IBM WebSphere Application Server V8.0'
  features='core.feature,com.ibm.jre.6_64bit' id='com.ibm.websphere.PLG.v80'/>
```

Follow these guidelines:

- Include this feature only if you are installing on a 64-bit system; do not include it if you are installing on a 32-bit system.
- You must include one of the two options if you are installing on a 64-bit system.
- You cannot modify this installation later and change the selection.

Installing fix packs on the Web Server Plug-ins using the Installation Manager GUI

You can use Installation Manager to update the Web Server Plug-ins to a later version.

Before you begin

Make sure that the web-based or local service repository location is listed and checked or that the **Search service repositories during installation and updates** option is selected on the Repositories panel in your Installation Manager preferences. For more information on using service repositories with Installation Manager, read the Installation Manager Information Center.

About this task

Perform this procedure to use Installation Manager to update the Web Server Plug-ins.

Procedure

1. Start Installation Manager.
2. Click **Update**.

   Note: If you are prompted to authenticate, use the IBM ID and password that you use to access protected IBM software websites.
3. Select the package group to update.
4. Click **Next**.
5. Select the version to which you want to update under **Web Server Plug-ins for IBM WebSphere Application Server**.
6. Click **Next**.
7. Accept the terms in the license agreements, and click **Next**.
8. Review the summary information, and click **Update**.
   - If the installation is successful, the program displays a message indicating that installation is successful.
   - If the installation is not successful, click **View Log File** to troubleshoot the problem.
9. Click **Finish**.
10. Click **File > Exit** to close Installation Manager.

Uninstalling fix packs from the Web Server Plug-ins using the Installation Manager GUI

You can use Installation Manager to roll back the Web Server Plug-ins to an earlier version.

Before you begin

During the rollback process, Installation Manager must access files from the earlier version of the package. By default, these files are stored on your computer when you install a package. If you change the default setting or delete the files using the **Remove Saved Files** option, Installation Manager requires access to the repository that was used to install the earlier version.

About this task

Perform this procedure to use Installation Manager to roll back the Web Server Plug-ins to an earlier version.

Procedure

1. Stop all servers on the WebSphere Application Server installation that is being modified.
2. Start Installation Manager.
3. Click **Roll Back**.

   Note: If you are prompted to authenticate, use the IBM ID and password that you use to access protected IBM software websites.
4. Select the package group to roll back.
5. Click **Next**.
6. Select the version to which you want to roll back under **Web Server Plug-ins for IBM WebSphere Application Server**.
7. Click **Next**.
8. Review the summary information, and click **Roll Back**.
   - If the rollback is successful, the program displays a message indicating that the rollback is successful.
   - If the rollback is not successful, click **View Log File** to troubleshoot the problem.

9. Click **Finish**.
10. Click **File > Exit** to close Installation Manager.

Uninstalling Web Server Plug-ins using the GUI

Use the Installation Manager GUI to uninstall the Web Server Plug-ins.

Before you begin

Removing the plug-ins will break communication between any web servers and their associated application servers. Be sure to unconfigure any web and applications servers that are using these plug-ins before uninstalling them.

Procedure

1. Start Installation Manager.
2. Click **Uninstall**.
3. In the **Uninstall Packages** window, perform the following actions.
   a. Select **Web Server Plug-ins for IBM WebSphere Application Server** and the appropriate version.

      Note: If you are uninstalling the ILAN version of this product, select **Web Server Plug-ins for IBM WebSphere Application Server (ILAN)**.

   b. Click **Next**.
4. Review the summary information.
5. Click **Uninstall**.
   - If the uninstallation is successful, the program displays a message that indicates success.
   - If the uninstallation is not successful, click **View log** to troubleshoot the problem.
6. Click **Finish**.
7. Click **File > Exit** to close Installation Manager.

Uninstalling the Web Server Plug-ins silently

You can use Installation Manager to uninstall the Web Server Plug-ins silently.

Before you begin

Removing the plug-ins will break communication between any web servers and their associated application servers. Be sure to unconfigure any web and applications servers that are using these plug-ins before uninstalling them.

Optional: Perform or record the installation of Installation Manager and installation of the Web Server Plug-ins to a temporary installation registry on one of your systems so that you can use this temporary registry to record the uninstallation without using the standard registry where Installation Manager is installed.

Read the following for more information:
- "Installing the Web Server Plug-ins using the GUI" on page 70
- "Installing the Web Server Plug-ins silently" on page 74

About this task

Using Installation Manager, you can work with response files to uninstall the Web Server Plug-ins silently in a variety of ways. You can record a response file using the GUI as described in the following procedure, or you can generate a new response file by hand or by taking an example and modifying it.

Procedure

1. Optional: **Record a response file to uninstall the Web Server Plug-ins:** On one of your systems, perform the following actions to record a response file that will uninstall the Web Server Plug-ins:
   a. From a command line, change to the eclipse subdirectory in the directory where you installed Installation Manager.
   b. Start Installation Manager from the command line using the -record option.

      For example:
      - Windows **Administrator or non-administrator:**

```
IBMIM.exe -skipInstall "C:\temp\imRegistry"
  -record C:\temp\uninstall_response_file.xml
```

      - AIX HP-UX Linux Solaris **Administrator:**

```
./IBMIM -skipInstall /var/temp/imRegistry
  -record /var/temp/uninstall_response_file.xml
```

      - AIX HP-UX Linux Solaris **Non-administrator:**

```
./IBMIM -skipInstall user_home/var/temp/imRegistry
  -record user_home/var/temp/uninstall_response_file.xml
```

      Tip: If you choose to use the -skipInstall parameter with a temporary installation registry created as described in "Before you begin," Installation Manager uses the temporary installation registry while recording the response file. It is important to note that when the -skipInstall parameter is specified, no packages are installed or uninstalled. All of the actions that you perform in Installation Manager simply update the installation data that is stored in the specified temporary registry. After the response file is generated, it can be used to uninstall the Web Server Plug-ins, removing the Web Server Plug-ins files and updating the standard installation registry.

      The -skipInstall operation should not be used on the actual agent data location used by Installation Manager. This is unsupported. Use a clean writable location, and re-use that location for future recording sessions.

      For more information, read the IBM Installation Manager Information Center.
   c. Click **Uninstall**.
   d. In the **Uninstall Packages** window, perform the following actions.
      1) Select **Web Server Plug-ins for IBM WebSphere Application Server** and the appropriate version.

         Note: If you are uninstalling the ILAN version of this product, select **Web Server Plug-ins for IBM WebSphere Application Server (ILAN)**.
      2) Click **Next**.
   e. Review the summary information.
   f. Click **Uninstall**.
      - If the uninstallation is successful, the program displays a message that indicates success.
      - If the uninstallation is not successful, click **View log** to troubleshoot the problem.
   g. Click **Finish**.
   h. Click **File > Exit** to close Installation Manager.
2. **Use the response file to uninstall the Web Server Plug-ins silently:** From a command line on each of the systems from which you want to uninstall the Web Server Plug-ins, change to the `eclipse/tools` subdirectory in the directory where you installed Installation Manager and use the response file that you created to silently uninstall the Web Server Plug-ins.

   For example:
   - Windows **Administrator or non-administrator:**

```
imcl.exe
  input C:\temp\uninstall_response_file.xml
  -log C:\temp\uninstall_log.xml
```

- AIX HP-UX Linux Solaris **Administrator:**

```
./imcl
  input /var/temp/uninstall_response_file.xml
  -log /var/temp/uninstall_log.xml
```

- AIX HP-UX Linux Solaris **Non-administrator:**

```
./imcl
  input user_home/var/temp/uninstall_response_file.xml
  -log user_home/var/temp/uninstall_log.xml
```

Go to the IBM Installation Manager Information Center for more information.

Windows
Example

The following is an example of a response file for silently uninstalling the Web Server Plug-ins.

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- ##### Copyright ######################################################
# Licensed Materials - Property of IBM (c) Copyright IBM Corp. 2011.
# All Rights Reserved. US Government Users Restricted Rights-Use, duplication
# or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
##################################################################### -->

<!-- ##### Frequently Asked Questions ####################################
# The latest information about using Installation Manager is
# located in the online Information Center. There you can find
# information about the commands and attributes used in
# silent installation response files.
#
#     Installation Manager Information Center can be found at:
#     http://publib.boulder.ibm.com/infocenter/install/v1r4/index.jsp
#
# Question 1. How do I record a response file using Installation Manager?
# Answer 1. Start Installation Manager from the command line under the
# eclipse subdirectory with the record parameter and it will generate a
# response file containing actions it performed, repositories it used, and
# its preferences settings. Optionally use the -skipInstall parameter if
# you do not want the product to be installed to the machine. Specify a
# new agentDataLocation location value when doing a new installation. Do
# not use an existing agentDataLocation for an installation because it might
# damage the installation data and prevent you from modifying, updating,
# rolling back, or uninstalling the installed packages.
#
# Windows: IBMIM -record <responseFile> -skipInstall <agentDataLocation>
# Linux or UNIX: ./IBMIM -record <responseFile> -skipInstall <agentDataLocation>
#
# For example:
#   Windows = IBMIM.exe -record c:\temp\responsefiles\WASv8.install.Win32.xml
#     -skipInstall c:\temp\skipInstall\WebSphere_Temp_Registry
#   Linux or UNIX = ./IBMIM -record /home/user/responsefiles/WASv8.install.RHEL64.xml
#     -skipInstall c:\temp\skipInstall\WebSphere_Temp_Registry
#
# Question 2. How do I run Installation Manager silently using response file?
# Answer 2. Create a silent installation response file and run the following command
# from the eclipse\tools subdirectory in the directory where you installed
# Installation Manager:
#
#   Windows = imcl.exe -acceptLicense -showProgress
#     input <response_file_path_and_name> -log <log_file_path_and_name>
#   Linux, UNIX, IBM i and z/OS = ./imcl -acceptLicense -showProgress
#     input <response_file_path_and_name> -log <log_file_path_and_name>
#
# For example:
#   Windows = imcl.exe -acceptLicense -showProgress
#     input c:\temp\responsefile\WASv8.install.Win32.xml
#   Linux, UNIX, IBM i and z/OS = ./imcl -acceptLicense -showProgress
#     input /home/user/responsefile/WASv8.install.RHEL64.xml
#
# The -acceptLicense command must be included to indicate acceptance of all
#     license agreements of all offerings being installed, updated or modified.
# The -showProgress command shows progress when running in silent mode.
# Additional commands can be displayed by requesting help:  IBMIM -help
#
##################################################################### -->

<!-- ##### Agent Input ###############################################
# The clean and temporary attributes specify the repositories and other
# preferences Installation Manager uses and whether those settings
# should persist after the uninstall finishes.
#
# Valid values for clean:
#      true = only use the repositories and other preferences that are
#           specified in the response file.
#      false = use the repositories and other preferences that are
#           specified in the response file and Installation Manager.
#
# Valid values for temporary:
#      true = repositories and other preferences specified in the
#           response file do not persist in Installation Manager.
#      false = repositories and other preferences specified in the
#           response file persist in Installation Manager.
#
##################################################################### -->

<agent-input clean='true' temporary='true'>

<!-- ##### Repositories ###############################################
# Repositories are locations that Installation Manager queries for
# installable packages. Repositories can be local (on the machine
# with Installation Manager) or remote (on a corporate intranet or
# hosted elsewhere on the internet).
#
# If the machine using this response file has access to the internet,
# then include the IBM WebSphere Live Update Repositories in the list
# of repository locations.
#
# If the machine using this response file cannot access the internet,
# then comment out the IBM WebSphere Live Update Repositories and
# specify the URL or UNC path to custom intranet repositories and
# directory paths to local repositories to use.
#
##################################################################### -->

<server>
    <!-- ##### IBM WebSphere Live Update Repositories ###################
     # These repositories contain Web Server Plug-ins for WebSphere
  # Application Server offerings, and updates for those offerings
     #
     # To use the secure repository (https), you must have an IBM ID,
     # which can be obtained by registering at: http://www.ibm.com/account
     # or your Passport Advantage account.
     #
     # And, you must use a key ring file with your response file.
     ################################################################ -->
    <repository location="http://www.ibm.com/software/repositorymanager/ccom.ibm.websphere.PLG.v80" />
    <!-- <repository location="https://www.ibm.com/software/rational/repositorymanager/repositories/websphere" /> -->

    <!-- ##### Custom Repositories #######################################
     # Uncomment and update the repository location key below
     # to specify URLs or UNC paths to any intranet repositories
     # and directory paths to local repositories to use.
     ################################################################ -->
    <!-- <repository location='https:\\w3.mycompany.com\repositories\'/> -->
    <!-- <repository location='/home/user/repositories/websphere/'/> -->

    <!-- ##### Local Repositories ########################################
     # Uncomment and update the following line when using a local
     # repository located on your own machine to install a
     # Web Server Plug-ins offering.
     ############################################################ -->
    <!-- <repository location='insert the full directory path inside single quotes'/> -->
</server>

<!-- ##### Uninstall Packages ##########################################
#
# Uninstall Command
#
# Use the uninstall command to inform Installation Manager of the
# installation packages to uninstall.
#
# The modify attribute is optional and can be paired with an install
# command to add features or paired with an uninstall command to
# remove commands. If omitted, the default value is set to false.
#    false = indicates not to modify an existing install by adding
#            or removing features.
#    true = indicates to modify an existing install by adding or
#           removing features.
#
# The offering ID attribute is required because it specifies the
# offering to be uninstalled. The example command below contains the
# offering ID for Web Server Plug-ins.
#
# The version attribute is optional. If a version number is provided,
# then the offering will be uninstalled at the version level specified
# If the version attribute is not provided, then the default behavior is
# to uninstall the latest version. The version number can be found in
# the repository.xml file in the repositories.
# For example, <offering ... version='8.0.0.20110617_2222'>.
#
# The profile attribute is required and must match the package group
# name for the offering to be uninstalled.
#
# The features attribute is optional. If there is no feature attribute,
# then all features are uninstalled. If features are specified, then
# only those features will be uninstalled.
# Features must be comma delimited without spaces.
#
# The feature values for Web Server Plug-ins include:
#  com.ibm.jre.6_32bit,com.ibm.jre.6_64bit
#
# Installation Manager supports uninstalling multiple offerings at once.
# Additional offerings can be included in the uninstall command,
# with each offering requiring its own offering ID, version, profile value,
# and feature values.
#
# Profile Command
#
# A separate profile command must be included for each offering listed
# in the install command. The profile command informs Installation
# Manager about offering specific properties or configuration values.
#
# The installLocation specifies where the offering will be installed.
# If the response file is used to modify or update an existing
# installation, then ensure the installLocation points to the
# location where the offering was installed previously.
#
# The eclipseLocation data key should use the same directory path to
# Web Server Plug-ins as the installationLocation attribute.
#
# Include data keys for product specific profile properties.
#
################################################################## -->
<uninstall modify='false'>
<offering id='com.ibm.websphere.PLG.v80'
 profile='Web Server Plug-ins for IBM WebSphere Application Server V8.0'
 features='core.feature,com.ibm.jre.6_32bit'/>
</uninstall>

<profile id='Web Server Plug-ins for IBM WebSphere Application Server V8.0'
 installLocation='C:\Program Files\IBM\WebSphere\Plugins'>
<data key='eclipseLocation' value='C:\Program Files\IBM\WebSphere\Plugins'/>
<data key='user.import.profile' value='false'/>
<data key='cic.selector.nl' value='en'/>
</profile>

<!-- ##### Shared Data Location ########################################
# Uncomment the preference for eclipseCache to set the shared data
# location the first time you use Installation Manager to do an
# installation.
#
# Eclipse cache location can be obtained from the installed.xml file found in
# Linux/Unix: /var/ibm/InstallationManager
# Windows: C:\Documents and Settings\All Users\Application Data\IBM\Installation Manager
# from the following property:
# <property name='cacheLocation' value='C:\Program Files\IBM\IMShared'/>
#
# Open the installed.xml file in a text editor because the style sheet
# might hide this value if opened in a web browser.
# For further information on how to edit preferences, refer to the public library at:
# http://publib.boulder.ibm.com/infocenter/install/v1r4/index.jsp?topic=/com.ibm.silentinstall12.doc/topics/r_silent_prefs.html
#
# After the shared data location is set, it cannot be changed
# using a response file or the graphical wizard.
#
# Ensure that the shared data location is a location that can be written
# to by all user accounts that are expected to use Installation Manager.
#
# By default, Installation Manager saves downloaded artifacts to
# the shared data location. This serves two purposes.
#
# First, if the same product is installed a more than once to the machine,
# then the files in the shared data location will be used rather than
# downloading them again.
#
# Second, during the rollback process, the saved artifacts are used.
# Otherwise, if the artifacts are not saved or are removed, then
# Installation Manager must have to access the repositories used to
# install the previous versions.
#
# Valid values for preserveDownloadedArtifacts:
#     true = store downloaded artifacts in the shared data location
#     false = remove downloaded artifacts from the shared data location
#
################################################################## -->

<!--
<preference name='com.ibm.cic.common.core.preferences.eclipseCache' value='C:\Program Files\IBM\IMShared'/>
<preference name='com.ibm.cic.common.core.preferences.preserveDownloadedArtifacts' value='true'/>
-->

<!-- ##### Preferences Settings ########################################
# Additional preferences for Installation Manager can be specified.
# These preference correspond to those that are located in the graphical
# interface under File / Preferences.
```

```
#
# If a preference command is omitted from or commented out of the response
# file, then Installation Manager uses the preference value that was
# previously set or the default value for the preference.
#
# Preference settings might be added or deprecated in new versions of
# Installation Manager. Consult the online Installation Manager
# Information Center for the latest set of preferences and
# descriptions about how to use them.
#
# http://publib.boulder.ibm.com/infocenter/install/v1r4/index.jsp
#
################################################################### -->

<!--
<preference name='com.ibm.cic.common.core.preferences.connectTimeout' value='30'/>
<preference name='com.ibm.cic.common.core.preferences.readTimeout' value='45'/>
<preference name='com.ibm.cic.common.core.preferences.downloadAutoRetryCount' value='0'/>
<preference name='offering.service.repositories.areUsed' value='true'/>
<preference name='com.ibm.cic.common.core.preferences.ssl.nonsecureMode' value='false'/>
<preference name='com.ibm.cic.common.core.preferences.http.disablePreemptiveAuthentication' value='false'/>
<preference name='http.ntlm.auth.kind' value='NTLM'/>
<preference name='http.ntlm.auth.enableIntegrated.win32' value='true'/>
<preference name='com.ibm.cic.common.core.preferences.keepFetchedFiles' value='false'/>
<preference name='PassportAdvantageIsEnabled' value='false'/>
<preference name='com.ibm.cic.common.core.preferences.searchForUpdates' value='false'/>
<preference name='com.ibm.cic.agent.ui.displayInternalVersion' value='false'/>
 -->

</agent-input>
```

Plug-ins configuration

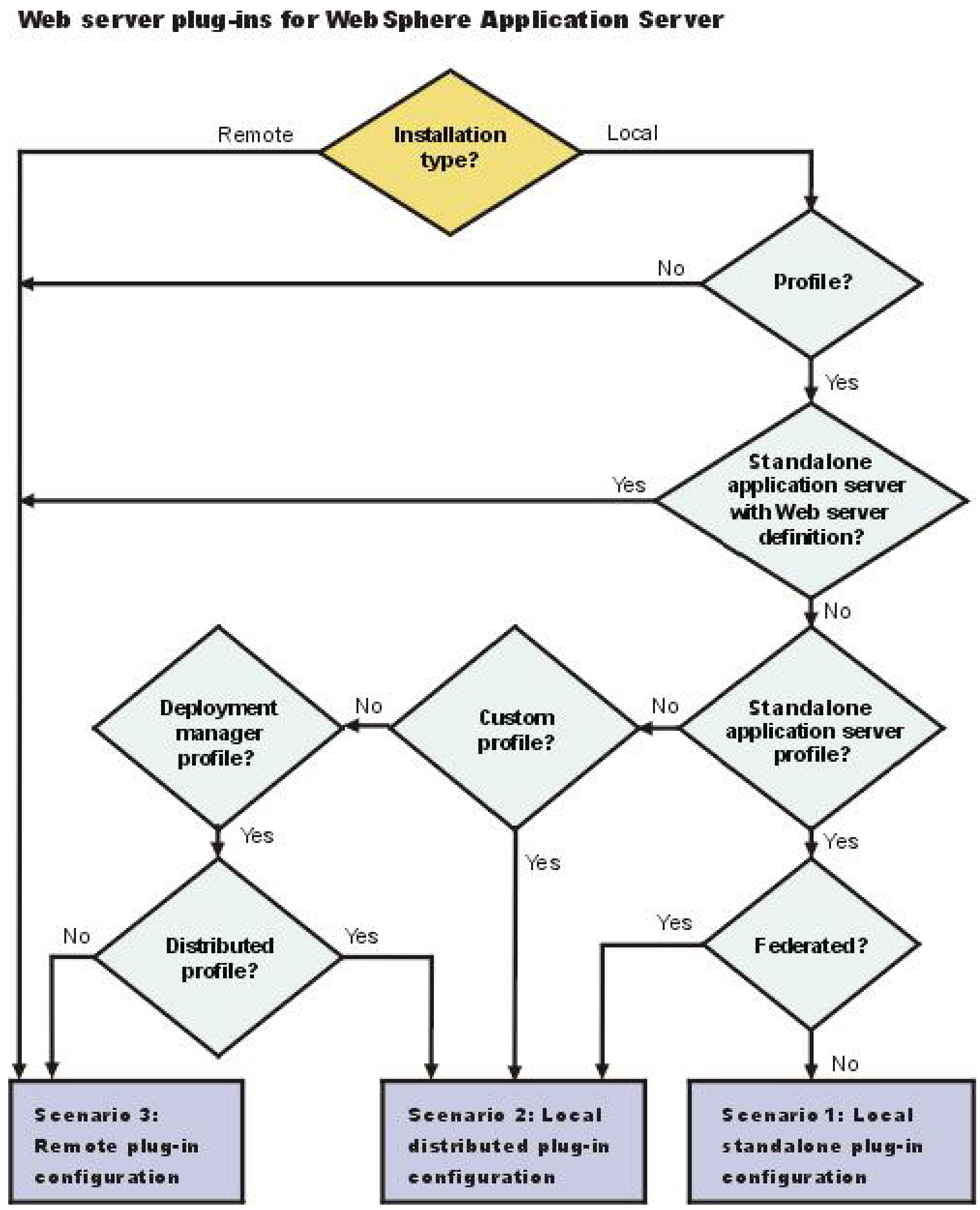
The Web Server Plug-ins Configuration Tool configures an application server for a web server type and creates a web server definition in the configuration of the application server. This overview shows the different processing paths that the Web Server Plug-ins Configuration Tool can use.

This topic describes the three ways that the Web Server Plug-ins Configuration Tool can configure a web server and create the `plugin-cfg.xml` file, which is the plug-in configuration file.

Tip: As an alternative to using the Web Server Plug-ins Configuration Tool, you can use the pct command-line tool with a response file to configure a web server. Read "Configuring a web server plug-in using the WCT command-line utility" on page 132 for more information.

Configuration flows for the Network Deployment product

The Web Server Plug-ins Configuration Tool resolves all configurations of a web server and WebSphere Application Server to three scenarios: a remote application server, local distributed application server, and local standalone application server. The logic implemented in determining which scenario applies to a configuration is shown in the following diagram.


Web server plug-ins for WebSphere Application Server
Remote
Installation type?
Local
No
Profile?
Yes
Yes
Standalone application server with Web server definition?
No
Deployment manager profile?
No
Custom profile?
No
Standalone application server profile?
Yes
Yes
Yes
No
Distributed profile?
Yes
Yes
Federated?
No
Scenario 3: Remote plug-in configuration
Scenario 2: Local distributed plug-in configuration
Scenario 1: Local standalone plug-in configuration

Legend:

Installation type?
The installation type is either remote or local.

When the Web server and application server are not on the same computer, choose the remote scenario. When both the web server and application server are on the same computer, choose the local scenario.

Profile?
If the product is installed but the Profile Management Tool has not yet created a profile, the scenario is considered to be a remote installation.

Standalone application server with web server definition?
If the profile is an application server with an existing web server definition, the installation is considered a remote installation.

Profile_type?
The Web Server Plug-ins Configuration Tool can configure only one profile at a time. These three paths show how processing varies for different types of profiles.

Federated?
If the application server node is federated, the Web Server Plug-ins Configuration Tool configures the web server definition on the managed node. This has advantages. Suppose the web server and the managed node are on a separate machine. The `plugin-cfg.xml` file is automatically propagated to the remote node during node synchronization because the web server definition is part of the node configuration.

Distributed profile?
If the deployment manager has a federated custom node (custom profile), the Web Server Plug-ins Configuration Tool configures the web server definition on the managed node. This has advantages. Suppose the web server and the managed node are on a separate machine. The `plugin-cfg.xml` file is automatically propagated to the remote node during node synchronization because the web server definition is part of the node configuration.

Scenario 1. Local standalone plug-in configuration

The Web Server Plug-ins Configuration Tool creates a web server definition within the application server profile.

The Web Server Plug-ins Configuration Tool configures the web server to use the `plugin-cfg.xml` file that is within the application server profile. The standalone application server regenerates the *profile_root*/`config/cells/`*cell_name*/`nodes/`*web_server_name*_`node/servers/`*web_server_name*/`plugin-cfg.xml` file whenever a change occurs in the application server configuration that affects deployed applications.

After installing the binary plug-in for the local web server, you can start the application server and the web server immediately upon completion of the installation.

Suppose that you create a web server definition on a standalone application server and then federate the node. The web server definition is not federated into the cell because the web server definition is defined as a separate node in a standalone application server. You must recreate the web server definition on the managed node. See **Scenario 2**.

Table 10. Configuration that qualifies for the local standalone application server scenario.

Profile type	Federation status	Automatic creation of web server definition?	Web server already defined in application server configuration?
Application server	Not federated	Yes	No

Redirection to Scenario 3

An unfederated standalone application server that has an existing web server definition should be processed as a remote plug-in configuration.

An existing web server definition on a standalone application server requires that the Web Server Plug-ins Configuration Tool follow the remote installation path. A standalone application server can have just one web server definition.

See **Scenario 3** for a description of this type of node.

Redirection to Scenario 2

A federated standalone application server should be processed as a local distributed plug-in configuration. See **Scenario 2** for a description of this type of node.

Overview of the verification procedure

The following overview shows the procedure for verifying the web server configuration:

1. Start the web server with the proper procedure for your web server.

   For example, start the IBM HTTP Server from a command line:

   - AIX HP-UX Linux Solaris `./`*`IHS_root`*`/bin/apachectl start`
   - Windows *IHS_root*`\bin\apache`
2. Start the application server.

   Change directories to the *profile_root*`/bin` directory and run the startServer command:

   - AIX HP-UX Linux Solaris `./`*`profile_root`*`/bin/startServer.sh server1`
   - Windows *profile_root*`\bin\startServer server1`

   Open the administrative console and save the changed configuration.
3. Point your browser to `http://localhost:9080/snoop` to test the internal HTTP transport provided by the application server. Point your browser to `http://`*`Host_name_of_Web_server_machine`*`/snoop` to test the web server plug-in.
4. Verify that both web addresses display the Snoop Servlet - Request/Client Information page.

Scenario 2. Local distributed plug-in configuration

The Web Server Plug-ins Configuration Tool does not automatically create a web server definition within a federated application server profile. The tool creates the `configure`*`web_server_name`* script instead in the *plugins_root*`/bin` directory.

The Web Server Plug-ins Configuration Tool configures the web server to use the `plugin-cfg.xml` file that will be created within the application server profile when you run the script. The deployment manager regenerates the `plugin-cfg.xml` file in the *profile_root*`/config/cells/`*cell_name*`/nodes/`*node_name*`/servers/`*web_server_name* directory. Regeneration occurs whenever a change occurs in the application server configuration that affects deployed applications on the managed node.

After installing the binary plug-in for the local web server, you must run the script before you can start the web server. The web server has already been configured to use the `plugin-cfg.xml` file in the application server configuration. That file does not exist until you run the `configure`*`web_server_name`* script.

Table 11. Configurations that qualify for the local distributed application server scenario.

Profile type	Federation status	Creation of web server definition?	Web server already defined in application server configuration?
Application server profile	Federated	By script	N/A
Custom profile	Not federated	By script	N/A
Custom profile	Federated	By script	N/A
Deployment manager profile with a managed node (distributed profile)	N/A	By script	N/A

The following overview shows the procedure for completing the configuration and verifying the web server configuration:

1. Start the deployment manager.
2. If you are planning to add an application server node into a deployment manager cell but have not done so yet, federate the node before installing the plug-ins. If the web server definition exists when you federate the node, the web server definition is lost when you federate.
3. Create the web server definition in the application server. You have two options:

- Use the administrative console of the deployment manager to create a web server definition for a managed node. Click **Servers > Web servers > New** and use the Create new web server entry wizard to create the web server definition.
- Run the script to manually create the web server definition within the configuration of the deployment manager. Run the script from the *plugins_root*/`bin` directory. The script can address the deployment manager on the same machine.

  Open a command window to run the appropriate script:
  - AIX HP-UX Linux Solaris `./configure`*`web_server_name`*`.sh`
  - Windows `configure`*`web_server_name`*`.bat`

  Note: The *webserverNodeName* value in the script is a concatenation of the nick name you have chosen for the web server and the suffix *-node*. It is automatically created during plug-in installation and cannot be changed. For example, if you named your web server *myserver* during plug-in installation, the value for the associated web server definition created after you ran the script would be *myserver-node*.

  If you have enabled security or changed the default JMX connector type, edit the script and include the appropriate parameters.

4. Start the web server with the proper procedure for your web server.

   For example, start the IBM HTTP Server from a command line:
   - AIX HP-UX Linux Solaris `./`*`IHS_root`*`/bin/apachectl start`
   - Windows *`IHS_root`*`\bin\apache`
5. Start the application server.

   Change directories to the *profile_root*/`bin` directory and run the startServer command:
   - AIX HP-UX Linux Solaris `./`*`profile_root`*`/bin/startServer.sh server1`
   - Windows *`profile_root`*`\bin\startServer server1`
6. Open the administrative console of the deployment manager. Wait for node synchronization to occur and save the changed configuration that includes the new web server definition.
7. Point your browser to `http://localhost:9080/snoop` to test the internal HTTP transport provided by the application server. Point your browser to `http://`*`Host_name_of_Web_server_machine`*`/snoop` to test the web server plug-in.
8. Verify that both web addresses display the Snoop Servlet - Request/Client Information page.

Scenario 3. Remote plug-in configuration

The Web Server Plug-ins Configuration Tool does not automatically create a web server definition within the distributed profile on a remote machine. The tool creates the `configure`*`web_server_name`* script instead.

The Web Server Plug-ins Configuration Tool configures the web server to use the `plugin-cfg.xml` file that will be maintained on the web server machine in the *plugins_root*/`config`/*`web_server_name`* directory. This file requires periodic propagation. Propagation is copying the current `plugin-cfg.xml` file from the application server machine to replace the *plugins_root*/`config`/*`web_server_name`*/`plugin-cfg.xml` file.

After installing the binary plug-in for the local web server, you do not have to run the script before you can start the application server and the web server. However, you do not have the benefits of a Web server definition in the application server node until you run the script.

Table 12. Configurations that qualify for the remote application server scenario.

Profile type	Federation status	Creation of web server definition?	Web server already defined in application server configuration?
Any profile anywhere if you select a remote installation type in the Web Server Plug-ins Configuration Tool	N/A	By script	N/A
No profile	N/A	By script	N/A

Table 12. Configurations that qualify for the remote application server scenario (continued).

Profile type	**Federation status**	**Creation of web server definition?**	**Web server already defined in application server configuration?**
Unfederated standalone application server profile with an existing web server definition	Not federated	By script	Yes
Deployment manager profile with no managed nodes	N/A	By script	N/A

Testing the application server without a web server definition: The following overview shows the procedure for verifying the temporary *plugins_root*/`config/`*`web_server_name`*`/plugin-cfg.xml` file.

The web server communicates with the remote application server using the temporary `plugin-cfg.xml` file.

If the application server has an HTTP Transport port assignment other than 9080, the test is not successful. Continue to the next section to create the web server definition on the application server and complete your test of the configuration.

1. Start the web server with the proper procedure for your web server.

   For example, start the IBM HTTP Server from a command line:

   - AIX HP-UX Linux Solaris `./`*`IHS_root`*`/bin/apachectl start`
   - Windows *IHS_root*`\bin\apache`
2. Start the application server on the remote machine.

   Change directories to the *profile_root*/`bin` directory and run the startServer command:

   - AIX HP-UX Linux Solaris `./`*`profile_root`*`/bin/startServer.sh server1`
   - Windows *`profile_root`*`\bin\startServer server1`
3. Point your browser to `http://localhost:9080/snoop` to test the internal HTTP transport provided by the application server. Point your browser to `http://`*`Host_name_of_Web_server_machine`*`/snoop` to test the web server plug-in.
4. Verify that both web addresses display the Snoop Servlet - Request/Client Information page.

Completing the installation by configuring a web server definition: The following overview shows the procedure for completing the configuration. The configuration is not complete until the Web server definition exists in the configuration of the application server node. The web server definition is a central element in the regeneration of a valid plug-in configuration file, `plugin-cfg.xml`.

1. Start the deployment manager if you are configuring the deployment manager or a managed node.
2. Federate a remote application server node or custom node now if you are planning to federate the node at some point. If a web server definition already exists when you federate a node, the definition is lost.
3. Create the web server definition in the application server. You have two options for a managed node. Use the script option for a deployment manager node without managed nodes.
   - Use the administrative console of the deployment manager to create a web server definition for a managed node. Click **Servers > Web servers > New** and use the Create new web server entry wizard to create the web server definition.
   - Run the script to manually create the web server definition within the configuration of the application server node:
     a. Copy the script from the *plugins_root*/`bin` directory to the remote *app_server_root*/`bin` directory.
     b. Open a command window and run the script:
        – AIX HP-UX Linux Solaris `./configure`*`web_server_name`*`.sh`
        – Windows `configure`*`web_server_name`*`.bat`

   Note: The *webserverNodeName* value in the script is a concatenation of the nick name you have chosen for the web server and the suffix *-node*. It is automatically created during plug-in installation and cannot be changed. For example, if you named your web server *myserver* during plug-in installation, the value for the associated web server definition created after you ran the script would be *myserver-node*.

If you have enabled security or changed the default Java Management Extensions (JMX) connector type, edit the script and include the appropriate parameters.

4. Open the administrative console of the deployment manager if the node is federated. Wait for node synchronization to occur on the managed node, and save the changed configuration that includes the new Web server definition. If the remote node is not federated, open the administrative console of the application server and save the changed configuration.
5. Copy the current plug-in configuration file, `plugin-cfg.xml`, in the *profile_root*/`config/cells/`*`cell_name`*`/nodes/`*`web_server_name`*`_node/servers/`*`web_server_name`* directory. Paste the file on the web server machine to replace the temporary *plugins_root*/`config/`*`web_server_name`*`/plugin-cfg.xml` file. The IBM HTTP Server supports automatic propagation. Other web servers require manual propagation.
6. Start the web server with the proper procedure for your web server.
7. Point your browser to `http://localhost:9080/snoop` to test the internal HTTP transport provided by the application server. Point your browser to `http://`*`Host_name_of_Web_server_machine`*`/snoop` to test the web server plug-in.
8. Verify that both web addresses display the Snoop Servlet - Request/Client Information page.

Summary

Three scenarios exist for Web Server Plug-ins. Each scenario revolves around a unique location for the plug-in configuration file, `plugin-cfg.xml`. The application server generates the plug-in configuration file. The purpose of the file is to publish the location of all of the application server elements that are relevant to a web server. Such elements include applications, virtual hosts for serving applications, clusters, and cluster members for example.

If the web server cannot get to the file on the application server machine, you must take the file to the web server. That process is called "propagation." Propagation is reserved for the remote plug-in configuration scenario, which is **Scenario 3** in this topic.

In each of the local scenarios, the Web server can get to the `plugin-cfg.xml` file because it is on the same machine as the file. Two local scenarios exist because of two distinct locations for a local `plugin-cfg.xml` file.

The configuration scheme for WebSphere Application Server puts the plug-in configuration file in a web server definition that is either within a web server node or a managed node. The type of node is the difference between **Scenario 2** and **Scenario 1** in this topic. All **Scenario 2** configurations require the web server definition to exist within a managed application server node. All **Scenario 1** configurations have the web server definition within its own web server node.

Limited management options do not let you create or delete the one web server definition in the administrative console of a standalone application server. The inability of a standalone application server to create a web server definition is the basis for the configuration scripts created by the Web Server Plug-ins Configuration Tool. Without the scripts you could not easily create a web server definition on a standalone application server node.

The location of the `plugin-cfg.xml` file for each configuration described in this topic is shown in the following table:

Table 13. Plug-in configuration file locations.

Scenario	Profile type	Location of the `plugin-cfg.xml` file		
		plugins_root	*profile_root*: **within the managed node**	*profile_root*: **within the web server node**
1	Application server profile			X
2	Application server profile		X	
	Custom profile		X	
	Deployment manager profile with a managed node (distributed profile)		X	
3	Any profile anywhere if you select a remote installation type in the Web Server Plug-ins Configuration Tool	X		
	No profile	X		
	Unfederated (standalone) application server profile with an existing web server definition	X		
	Deployment manager profile with no managed nodes	X		

Legend:

plugins_root
plugins_root

`/config/`*`web_server_name`*`/plugin-cfg.xml`

profile_root: **within the managed node**

`profile_root``/config/cells/`*`cell_name`*`/nodes/`*`node_name_of_AppServer`*`/servers/`*`web_server_name`*`/plugin-cfg.xml`

profile_root: **within the web server node**

`profile_root``/config/cells/`*`cell_name`*`/nodes/`*`web_server_name`*`_node/servers/`*`web_server_name`*`/plugin-cfg.xml`

Configuring a web server and an application server profile on the same machine

This topic describes configuring a web server plug-in that WebSphere Application Server provides to communicate with a particular brand of web server. This procedure describes installing the web server and its web server plug-in for WebSphere Application Server and the application server on the same machine.

Before you begin

When multiple profiles exist, you can select the profile that the Web Server Plug-ins Configuration Tool configures. See "Plug-ins configuration" on page 91 for a description of the flow of logic that determines how to select the profile to configure.

If the WebSphere Application Server product family supports a particular brand of web server, such as IBM HTTP Server or Microsoft Internet Information Services (IIS), your WebSphere Application Server product provides a binary plug-in for the web server that you must install.

If the WebSphere Application Server product family does not provide a binary plug-in for a particular brand of web server, then the web server is not supported. The purpose of the binary plug-in is to provide the communication protocol between the web server and the application server.

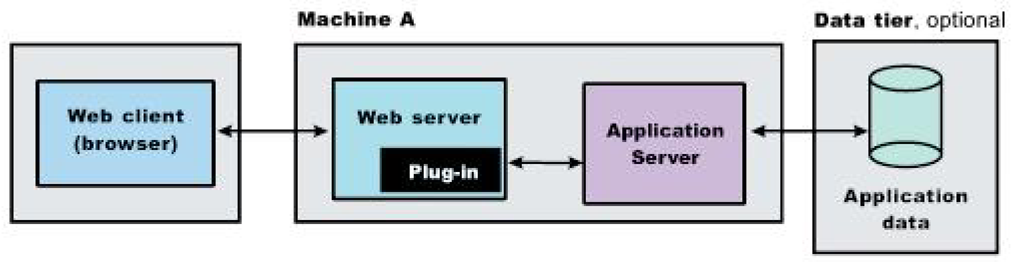
Suppose that you create a new profile and you also want to use a web server. You must install a new web server for the new profile, install the Web Server Plug-ins, and use the Web Server Plug-ins Configuration Tool to configure both the web server and the application server.

If the web server is not already installed, you can still install the Web Server Plug-ins for future use.

This procedure configures the application server profile that is the default profile on the machine. A one-to-one relationship exists between a web server and the application server.

However, a standalone application server profile and a managed profile can each have multiple web servers defined, each in a separate web server definition.

This article describes how to create the following topology:



Note: Nonroot installation for the plug-in component is only supported if the application server was also installed by the same nonroot user. Otherwise, the web server configuration scripts will fail to run against the application server installation.

About this task

Note: This topic references one or more of the application server log files. Beginning in WebSphere Application Server Version 8.0 you can configure the server to use the High Performance Extensible Logging (HPEL) log and trace infrastructure instead of using `SystemOut.log` , `SystemErr.log`, `trace.log`, and `activity.log` files or native z/OS logging facilities. If you are using HPEL, you can access all of your log and trace information using the LogViewer command-line tool from your server profile bin directory. See the information about using HPEL to troubleshoot applications for more information on using HPEL.

The Web Server Plug-ins Configuration Tool configures the plug-in for the supported web server after collecting the following information:

- Type of web server to configure
- Architecture of your installed target web server (64 bit or 32 bit)
- Location of the configuration file or files for the web server to be configured
- Web server port
- For IBM HTTP Server, the following information:
  - Port number for optional IBM HTTP Server administrative server setup
  - User ID and password to authenticate to the optional IBM HTTP Server administrative server from the administrative console
  - AIX HP-UX Linux Solaris System user ID and group to have write permission to IBM HTTP Server, the IBM HTTP Server administrative server, and the web server plug-in configuration files
  - Windows User ID and password if you choose to run the IBM HTTP Server administrative server as a Window service
- Name of the web server definition
- Configuration scenario to be used
  - If it is a remote scenario, the tool collects the host name or IP address of the application server.
  - If it is a local scenario, the tool collects the installation root directory of the WebSphere Application Server product.
- Profile to be configured with the web server plug-in

The Web Server Plug-ins Configuration Tool edits the configuration file or files for a web server by creating directives that point to the location of the binary plug-in module and the plug-in configuration file.

The name of the binary plug-in module varies per web server type. The plug-in configuration file is always the `plugin-cfg.xml` file.

The Web Server Plug-ins Configuration Tool creates a web server definition in the configuration of the application server unless one already exists.

You can use the administrative console to manage the web server configuration. For example, when you install an application on the application server, you can also choose to install it on the web server definition. If so, the updated `plugin-cfg.xml` file shows that the new application is available. When the web server reads the updated plug-in configuration file, the web server becomes aware of the new application that it can serve to web clients.

If you choose not to install the new application on the web server definition, the application is not added to the plug-in configuration file. The web server is not aware of the application and cannot serve it to web clients.

Tip: As an alternative to using the Web Server Plug-ins Configuration Tool, you can use the pct command-line tool with a response file to configure a web server. Read "Configuring a web server plug-in using the WCT command-line utility" on page 132 for more information.

Use this procedure to install the web server plug-in, configure the web server, and create a web server definition in the default application server profile.

Procedure

- Configure a standalone application server.
  1. Log on to the operating system.

     If you are installing as a nonroot or non-administrative user, then there are certain limitations.

     AIX HP-UX Linux Solaris In addition, select a umask that allows the owner to read/write to the files, and allows others to access them according to the prevailing system policy. For root, a umask of 022 is recommended. For nonroot users, a umask of 002 or 022 could be used, depending on whether or not the users share the group. To verify the umask setting, issue the following command:

```
umask
```

     To set the umask setting to 022, issue the following command:

```
umask 022
```

     Windows When installing as an administrative user on a Windows operating system, a Windows service is automatically created to autostart the application server. The installer user account must have the following advanced user rights:
     - Act as part of the operating system
     - Log on as a service

     For example, on some Windows operating systems, click **Control Panel > Administrative Tools > Local Security Policy > Local Policies > User Rights Assignments** to set the advanced options. See your Windows operating system documentation for more information.

     Windows If you plan to run the application server as a Windows service, do not install from a user ID that contains spaces. A user ID with spaces cannot be validated. Such a user ID is not allowed to continue the installation. To work around this restriction, install with a user ID that does not contain spaces.
  2. Install Installation Manager.
  3. Use Installation Manager to install the following:

   - WebSphere Application Server Network Deployment
   - Web Server Plug-ins for WebSphere Application Server
   - Websphere Customization Toolbox
4. Use Installation Manager to install the IBM HTTP Server, or install another supported web server.
5. Open the Websphere Customization Toolbox, and launch the Web Server Plug-ins Configuration Tool.
6. Select a Web Server Plug-ins runtime location.

   If the location of a previously installed web server plug-in that you want to use is not in the list, perform the following actions to add the location to your working set:
   a. Click **Add**.
   b. Enter a name for the web server plug-in location.
   c. Perform one of the following actions:
      - Enter the location.
      - Click **Browse**, find the location, and click **OK**.
7. Click **Create**.
8. Select the type of web server that you are configuring, and click **Next**.
9. Select the architecture of your installed target web server (64 bit or 32 bit) and click **Next** if you are asked.
10. Click **Browse** to select the configuration file or files for your web server, verify that the web server port is correct, and then click **Next** when you are finished.

    Select the file and not just the directory of the file. Some web servers have two configuration files and require you to browse for each file.

    The following list shows configuration files for supported web servers:

    Apache HTTP Server
    apache_root/`config/httpd.conf`

    Domino Web Server
    `names.nsf` and `Notes.jar`

    The wizard prompts for the `notes.jar` file. The actual name is `Notes.jar`.

    The Web Server Plug-ins Configuration Tool verifies that the files exist but the tool does not validate either file.

    IBM HTTP Server
    IHS_root/`conf/httpd.conf`

    Microsoft Internet Information Services (IIS)
    The Web Server Plug-ins Configuration Tool can determine the correct files to edit.

    Sun Java System Web Server (formerly Sun ONE Web Server and iPlanet Web Server) Version 6.0 and later
    `obj.conf` and `magnus.conf`
11. If you are configuring an IBM HTTP web server plug-in, perform the following actions.
    a. Optionally, set up the administration server configuration to administer the web server.
       1) Select **Setup IBM HTTP Server Administration Server**.
       2) Specify a port number on which the IBM HTTP administration server will communicate.
       3) Optionally, select **Create a user ID for IBM Server Administration Server authentication** and enter a user ID and password to authenticate to the IBM HTTP Server administrative server from the administrative console.
    b. Click **Next**.

c. AIX HP-UX Linux Solaris Specify the system user ID and group to have write permission to IBM HTTP Server, the IBM HTTP Server administrative server, and the web server plug-in configuration files.

   Select **Create a new unique system user ID and group using the credentials** if necessary.

d. Windows Optionally, set up the IBM HTTP Server Administration Server to run as a Window service.

   1) Select **Run IBM HTTP Server Administration Server as a Windows Service**.
   2) Perform one of the following actions:
      – Select **Log on as a local system account**.
      – Select **Log on as a specified user account**, and enter the user ID and password for that account.

        The user ID requires the following advanced user rights:
        - Act as part of the operating system
        - Log on as a service
   3) Choose whether your startup type will be automatic or manual.

e. Click **Next**.

12. Specify a unique name for the web server definition, and click **Next**.
13. Select the configuration scenario.
    a. Choose the local scenario.
    b. Perform one of the following actions:
       – Enter the installation location of WebSphere Application Server (*app_server_root*).
       – Click **Browse**, find the installation location of WebSphere Application Server (*app_server_root*), and click **OK**.
    c. Click **Next**.
14. Select the profile to configure with the current web server plug-in, and click **Next**.
15. Review the summary information, and click **Configure** to begin configuring the web server, web server plug-in, and application server profile.
16. Verify the success of the installation on the summary panel, and click **Finish**.

    If a problem occurs and the installation is unsuccessful, examine the logs in the *plugins_root*/`logs` directory. Correct any problems and re-configure.
17. **Domino Web Server only:** Set the WAS_PLUGIN_CONFIG_FILE environment variable.

    On platforms such as AIX or Linux, sourcing a script to the parent shell allows child processes to inherit the exported variables. On Windows systems, run the script as you would run any other command. Sourcing is automatic on Windows systems.
    a. Open a command window.
    b. Change directories to the plug-ins installation root directory.
    c. Issue the appropriate command for the *plugins_root*/`bin/setupPluginCfg.sh` script:
       – AIX HP-UX Solaris `. `*`plugins_root`*`/bin/setupPluginCfg.sh` (Notice the space between the period and the installation root directory.)
       – Linux `source `*`plugins_root`*`/bin/setupPluginCfg.sh`

    The script is also in the *lotus_root*/`notesdata` directory on operating systems such as AIX or Linux.

    Issue the appropriate command for the script before starting the Domino Web Server.
18. Start the Snoop servlet to verify the ability of the web server to retrieve an application from the application server.

    Test your environment by starting your application server, your web server, and using the Snoop servlet with an IP address.

a. Start the application server. In a Network Deployment environment, the Snoop servlet is available in the cell only if you included the DefaultApplication when adding the application server to the cell. The -includeapps option for the addNode command migrates the DefaultApplication to the cell. If the application is not present, skip this step.

   Change directories to the *profile_root*/`bin` directory and run the startServer command:

   – AIX HP-UX Linux Solaris `./startServer.sh server1`

   – Windows `startServer server1`

b. Start the IBM HTTP Server or the web server that you are using.

   Use a command window to change the directory to the IBM HTTP Server installed image, or to the installed image of your web server. Issue the appropriate command to start the web server, such as these commands for IBM HTTP Server:

   To start the IBM HTTP Server from the command line:

   Access the apache and apachectl commands in the `IBMHttpServer/bin` directory.

   – AIX HP-UX Linux Solaris `./apachectl start`

   – Windows `apache`

c. Point your browser to `http://localhost:9080/snoop` to test the internal HTTP transport provided by the application server. Point your browser to `http://`*`Host_name_of_Web_server_machine`*`/snoop` to test the web server plug-in.

   The HTTP Transport port is 9080 by default and must be unique for every profile. The port is associated with a virtual host named default_host, which is configured to host the installed DefaultApplication. The Snoop servlet is part of the DefaultApplication. Change the port to match your actual HTTP Transport port.

d. Verify that Snoop is running.

   Either web address should display the Snoop Servlet - Request/Client Information page.

   Tip: In the event of a verification failure where an HTTP error code of 500 appears, go to **IIS Manager > Default Web Site > sePlugins**. Right click, and choose to edit permissions. Click on the sharing tab, and choose to share with everyone (permissions level: read/write).

e. **Remote IBM HTTP Server only:**

   Verify that the automatic propagation function can work on a remote IBM HTTP Server by using the following steps. This procedure is not necessary for local web servers.

   1) Create a user=adminUser, password=adminPassword in the *IHS_root* `/conf/admin.passwd` file. For example: `c:\ws\ihs80\bin\htpasswd -cb c:\ws\ihs80\conf\admin.passwd adminUser adminPassword`

   2) Use the administrative console of the deployment manager or the application server to enter the User ID and password information that you created for the administrative user of IBM HTTP Server. Go to **Servers > Web server > *web_server_definition* > Remote web server administration**. Set the following values: admin Port=8008, User Id=adminUser, Password=adminPassword.

   3) Set the correct read/write permissions for the `httpd.conf` file and the plugin-cfg.xml file. See the *IHS_root* `/logs/admin_ERROR. LOG` file for more information.

   Automatic propagation of the plug-in configuration file requires the IBM HTTP administrative server to be up and running. If you are managing an IBM HTTP Server using the WebSphere Application Server administrative console, the following error might display:

```
"Could not connect to IHS Administration server error"
```

   Perform the following procedure to correct the error:

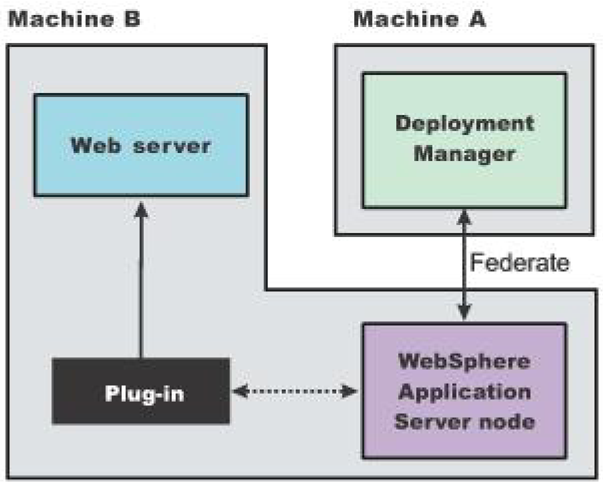
   1) Verify that the IBM HTTP Server administration server is running.

2) Verify that the web server host name and the port that is defined in the WebSphere Application Server administrative console matches the IBM HTTP Server administration host name and port.
3) Verify that the fire wall is not preventing you from accessing the IBM HTTP Server administration server from the WebSphere Application Server administrative console.
4) Verify that the user ID and password that is specified in the WebSphere Application Server administrative console under remote managed, is created in the `admin.passwd` file, using the htpasswd command.
5) If you are trying to connect securely, verify that you export the IBM HTTP Server administration server keydb personal certificate into the WebSphere Application Server key database as a signer certificate. This key database is specified by the com.ibm.ssl.trustStore directive in the `sas.client.props` file in the profile where your administrative console is running. This consideration is primarily for self-signed certificates.
6) If you still have problems, check the IBM HTTP Server `admin_ERROR. LOG` file and the WebSphere Application Server logs (`trace.log` file) to determine the cause of the problem.

- Configure an application server that is federated into a deployment manager cell.

Note: The following procedure describes installing the plug-ins on two machines. However, you could perform this procedure on a single machine.

The following topology is considered a local distributed topology because it involves a cell:



This part of the procedure assumes that you have already installed Installation Manager and the Network Deployment product on both machines. Also assumed is that you have already configured a deployment manager profile on Machine A and an application server profile on Machine B.

If you are planning to add the application server node into a deployment manager cell but have not done so yet, start the deployment manager and federate the node before configuring the plug-in. You cannot add an application server with a web server definition into the deployment manager cell.

A web server definition on a federated application server is installed on the same managed node as the application server. There is one node, but with two server processes, the application server and the web server definition.

If you are installing the plug-ins for use with a federated application server, start the deployment manager. Verify that the node agent process on the managed node is also running. Both the deployment manager and the node agent must be running to successfully configure a managed node.

1. Use Installation Manager to install the following on Machine B.
   – Web Server Plug-ins for WebSphere Application Server
   – Websphere Customization Toolbox

2. Use Installation Manager to install IBM HTTP Server on Machine B, or install another supported web server on Machine B.
3. Open the WebSphere Customization Toolbox and launch the Web Server Plug-ins Configuration Tool on Machine B.
4. Select a web server plug-in runtime location.

   If the location of a previously installed web server plug-in that you want to use is not in the list, perform the following actions to add the location to your working set:

   a. Click **Add**.
   b. Enter a name for the web server plug-in location.
   c. Perform one of the following actions:
      – Enter the location.
      – Click **Browse**, find the location, and click **OK**.
5. Click **Create**.
6. Select the type of web server that you are configuring, and click **Next**.
7. Select the architecture of your installed target web server (64 bit or 32 bit), and click **Next**.
8. Click **Browse** to select the configuration file or files for your web server, verify that the web server port is correct, and then click **Next** when you are finished.

   Select the file and not just the directory of the file. Some web servers have two configuration files and require you to browse for each file.

   The following list shows configuration files for supported web servers:

   Apache HTTP Server
   apache_root/`config/httpd.conf`

   Domino Web Server
   `names.nsf` and `Notes.jar`

   The wizard prompts for the `notes.jar` file. The actual name is `Notes.jar`.

   The web Server Plug-ins Configuration Tool verifies that the files exist but the tool does not validate either file.

   IBM HTTP Server
   IHS_root/`conf/httpd.conf`

   Microsoft Internet Information Services (IIS)
   The Web Server Plug-ins Configuration Tool can determine the correct files to edit.

   Note: The best practice is to use 32-bit plug-ins on IIS 7.

   Sun Java System Web Server (formerly Sun ONE Web Server and iPlanet Web Server) Version 6.0 and later
   `obj.conf` and `magnus.conf`
9. Specify a unique name for the web server definition, and click **Next**.
10. Select the configuration scenario.
    a. Choose the local scenario.
    b. Perform one of the following actions:
       – Enter the installation location of WebSphere Application Server (*app_server_root*).
       – Click **Browse**, find the installation location of WebSphere Application Server (*app_server_root*), and click **OK**.
    c. Click **Next**.
11. Select the profile to configure with the current web server plug-in, and click **Next**.
12. Review the summary information, and click **Configure** to begin configuring the web server, web server plug-in, and application server profile.

13. Verify the success of the installation on the summary panel, and click **Finish**.

    If a problem occurs and the installation is unsuccessful, examine the logs in the *plugins_root*/`logs` directory. Correct any problems and re-configure.
14. Copy the `configure`*`web_server_name`* script to paste on Machine A.
15. Create the web server definition on Machine A.

    You can use the administrative console of the deployment manager to create the web server definition on a federated node; or you can run the configuration script that the Web Server Plug-ins Configuration Tool created.

    The script already contains all of the information that you must gather when using the administrative console option.

    Select one of the following options:

    – **Using the administrative console**

      Click **Servers > Web servers > New** and use the Create new web server entry wizard to create the web server definition.

    – **Running the configuration script**

      a. Paste the `configure`*`web_server_name`* script from Machine B to the *`app_server_root`*`/bin` directory on Machine A.

      b. Issue the appropriate command from a command window:

         - AIX HP-UX Linux Solaris ./*`plugins_root`*`/bin/configure`*`web_server_name`*`.sh`
         - Windows *`plugins_root`*`\bin\configure`*`web_server_name`*`.bat`

      If you have enabled security or changed the default JMX connector type, edit the script and include the appropriate parameters on the wsadmin command.
16. From the administrative console of the deployment manager, click **System administration > Save Changes to Master Repository > Synchronize changes with Nodes > Save**.
17. **Domino web server only:** Set the WAS_PLUGIN_CONFIG_FILE environment variable.

    On platforms such as AIX or Linux, sourcing a script to the parent shell allows child processes to inherit the exported variables. On Windows systems, run the script as you would run any other command. Sourcing is automatic on Windows systems.
18. Start the Snoop servlet.

    See the Snoop procedure for the standalone application server for the full procedure.

Results

The installation of the Web Server Plug-ins results in the creation of the `Plugins` directory and several subdirectories. The following directories are among those created on a Linux system, for example:

- *plugins_root*/`bin` contains the binary plug-ins for all supported web servers
- *plugins_root*/`logs` contains log files
- *plugins_root*/`properties` contains version information

The Web Server Plug-ins Configuration Tool creates a web server definition within the application server profile unless one already exists.

The Web Server Plug-ins Configuration Tool configures the web server to use the *profile_root*/`plugin-cfg.xml` file.

The application server regenerates the web server plug-in configuration file, `plugin-cfg.xml` whenever an event occurs that affects the file. Such events include the addition or removal of an application, server, or virtual host. The standalone application server regenerates the file in the following location:

```
profile_root
   /config/cells/cell_name/nodes/
   web_server_name_node/servers/
   web_server_name/plugin-cfg.xml
```

On a federated node, the creation or removal of clusters and cluster members also causes file regeneration. The deployment manager regenerates the file for a federated application server in the following location:

```
profile_root
   /config/cells/cell_name/nodes/
   node_name_of_AppServer/servers/
   web_server_name/plugin-cfg.xml
```

What to do next

You can start a standalone application server and the web server immediately after configuring the plug-in for the local web server. Open the administrative console of the application server after you start the server and save the changed configuration.

After configuring the plug-in for the local web server, you can start a federated application server and the web server after running the script that completes the configuration. Open the administrative console of the deployment manager. Wait for node synchronization to occur. Save the changed configuration that includes the new web server definition.

See "Selecting a web server topology diagram and roadmap" on page 61 for an overview of the installation procedure.

See "Plug-ins configuration" on page 91 for information about the location of the plug-in configuration file.

See "Web server configuration" on page 39 for information about the files involved in configuring a web server.

See "Editing web server configuration files" on page 12 for information about how the Web Server Plug-ins Configuration Tool configures supported web servers.

See "Installing and configuring web server plug-ins" on page 58 for information about other installation scenarios for installing web server plug-ins.

Configuring a web server and an application server on separate machines (remote)

This topic describes installing a web server plug-in that WebSphere Application Server provides to communicate with a particular brand of web server. This procedure describes installing the web server and its web server plug-in for WebSphere Application Server on one machine and configuring the application server in the default profile on another machine to communicate with the web server.

Before you begin

When multiple profiles exist, you can select the profile that the Web Server Plug-ins Configuration Tool configures. See "Plug-ins configuration" on page 91 for a description of the flow of logic that determines how to select the profile to configure.

If the WebSphere Application Server product family supports a particular brand of web server, such as IBM HTTP Server or Microsoft Internet Information Services (IIS), your WebSphere Application Server product provides a binary plug-in for the web server that you must install.

If the WebSphere Application Server product family does not provide a binary plug-in for a particular brand of web server, then the web server is not supported. The purpose of the binary plug-in is to provide the communication protocol between the web server and the application server.

Suppose that you create a new profile and you also want to use a web server. You must install a new web server for the new profile, install the Web Server Plug-ins, and use the Web Server Plug-ins Configuration Tool to configure both the web server and the application server.

If the web server is not already installed, you can still install the Web Server Plug-ins for future use.

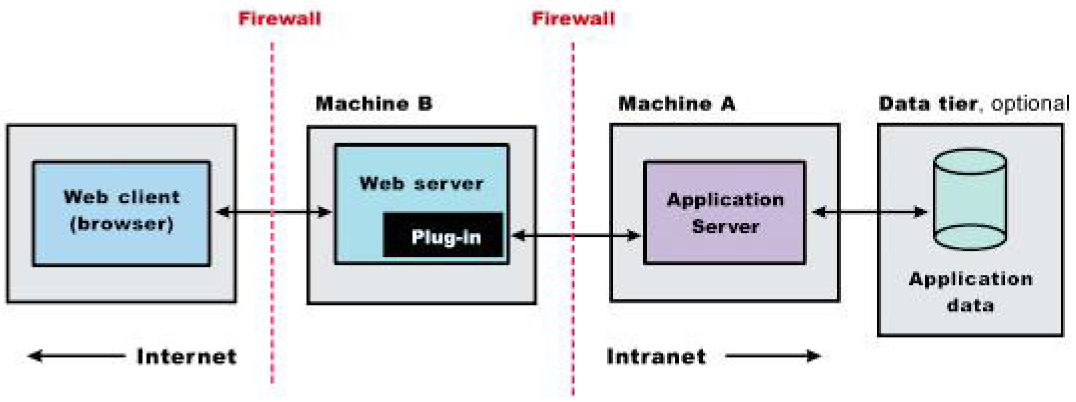
About this task

Installing the Web Server Plug-ins installs the plug-in module. The Web Server Plug-ins Configuration Tool configures the web server for communicating with the application server and creates a web server configuration definition in the application server, if possible.

Tip: As an alternative to using the Web Server Plug-ins Configuration Tool, you can use the pct command-line tool with a response file to configure a web server. Read “Configuring a web server plug-in using the WCT command-line utility” on page 132 for more information.

This procedure configures the application server profile that is the default profile on the machine. A one-to-one relationship exists between a web server and the application server.

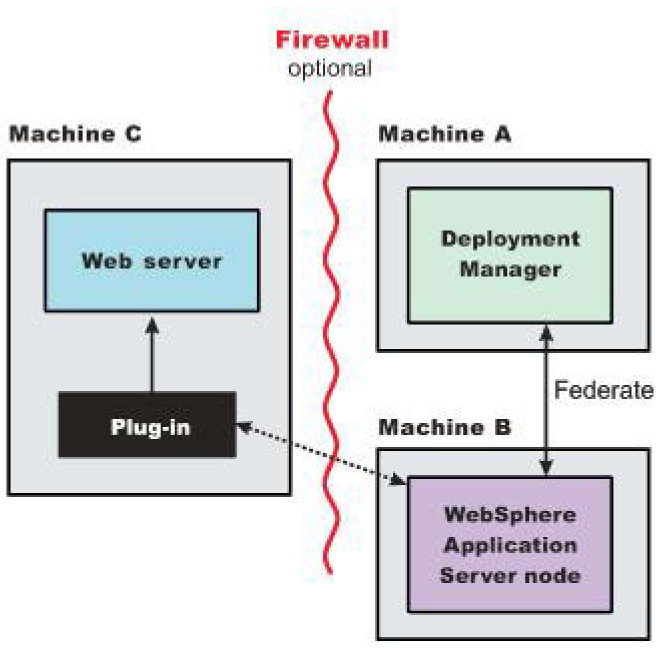
This article describes how to create the following topology:



Attention:

If you are planning to add the application server node into a deployment manager cell but have not done so yet, start the deployment manager and federate the node before configuring the plug-in. You cannot add an application server with a web server definition into the deployment manager cell.

The following topology is considered a remote topology because the web server is on a separate machine. The diagram shows a typical remote topology for a distributed environment:



This article describes the installation of a web server on one machine and the application server on a separate machine. In this situation, the Web Server Plug-ins Configuration Tool on one machine cannot create the web server definition in the application server configuration on the other machine.

In such a case, the Web Server Plug-ins Configuration Tool creates a script on the web server machine that you can copy to the application server machine. Run the script on the application server machine to create the web server configuration definition within the application server configuration.

Perform the following procedure to install the plug-in and configure both the web server and the application server.

Procedure

1. Install Installation Manager on Machine A and Machine B.
2. Use Installation Manager to install WebSphere Application Server Network Deployment on Machine A.
3. Create a standalone application server on Machine A.
4. Optional: Create a new host alias for the default virtual host.

   If you configured the web server to use a port other than port 80, then you must add a new host alias for that port for the default host. For example, when running as nonroot, IBM HTTP Server is configured with a default port value of 8080.
5. Use Installation Manager to install the following on Machine B.
   - Web Server Plug-ins for WebSphere Application Server
   - Websphere Customization Toolbox

6. Use Installation Manager to install the IBM HTTP Server on Machine B, or install another supported web server on Machine B.
7. Open the WebSphere Customization Toolbox, and launch the Web Server Plug-ins Configuration Tool on the machine with the web server.
8. Select a web server plug-in runtime location.

   If the location of a previously installed web server that you want to use is not in the list, perform the following actions to add the location to your working set:

   a. Click **Add**.
   b. Enter a name for the web server plug-in location.
   c. Perform one of the following actions:
      - Enter the location.
      - Click **Browse**, find the location, and click **OK**.
9. Click **Create**.
10. Select the type of web server that you are configuring, and click **Next**.
11. Select the architecture of your installed target web server (64 bit or 32 bit) and click **Next** if you are asked.
12. Click **Browse** to select the configuration file or files for your web server, verify that the web server port is correct, and then click **Next** when you are finished.

    Select the file and not just the directory of the file. Some web servers have two configuration files and require you to browse for each file.

    The following list shows configuration files for supported web servers:

    Apache HTTP Server
    apache_root/`config/httpd.conf`

    Domino Web Server
    `names.nsf` and `Notes.jar`

    The wizard prompts for the `notes.jar` file. The actual name is `Notes.jar`.

    The Web Server Plug-ins Configuration Tool verifies that the files exist but the tool does not validate either file.

    IBM HTTP Server
    IHS_root/`conf/httpd.conf`

    Microsoft Internet Information Services (IIS)
    The Web Server Plug-ins Configuration Tool can determine the correct files to edit.

    Sun Java System Web Server (formerly Sun ONE Web Server and iPlanet Web Server) Version 6.0 and later
    `obj.conf` and `magnus.conf`
13. If you are configuring an IBM HTTP web server plug-in, perform the following actions.

    a. Optionally, set up the administration server configuration to administer the web server.
       1) Select **Setup IBM HTTP Server Administration Server**.
       2) Specify a port number on which the IBM HTTP administration server will communicate.
       3) Optionally, select **Create a user ID for IBM Server Administration Server authentication** and enter a user ID and password to authenticate to the IBM HTTP Server administrative server from the administrative console.
    b. Click **Next**.
    c. AIX HP-UX Linux Solaris Specify the system user ID and group to have write permission to IBM HTTP Server, the IBM HTTP Server administrative server, and the web server plug-in configuration files.

       Select **Create a new unique system user ID and group using the credentials** if necessary.

d. Windows Optionally, set up the IBM HTTP Server Administration Server to run as a Window service.
   1) Select **Run IBM HTTP Server Administration Server as a Windows Service**.
   2) Perform one of the following actions:
      - Select **Log on as a local system account**.
      - Select **Log on as a specified user account**, and enter the user ID and password for that account.

        The user ID requires the following advanced user rights:
        – Act as part of the operating system
        – Log on as a service
   3) Choose whether your startup type will be automatic or manual.

e. Click **Next**.

14. Specify a unique name for the web server definition, and click **Next**.
15. Select the configuration scenario.
    a. Choose the remote scenario.
    b. Identify the host name or IP address of Machine A, which is the application server machine.
    c. Click **Next**.
16. Select the profile to configure with the current web server plug-in, and click **Next**.

    This panel does not display if you selected the remote scenario in the previous step.
17. Examine the summary panel, and click **Configure** to begin configuring.

    The panel notifies you that you have manual steps to perform to complete the installation and configuration.

    The Web Server Plug-ins Configuration Tool creates the `configure`*`web_server_name`* script in the *plugins_root*/`bin/` directory on Machine B (the machine with the web server).

    The Web Server Plug-ins Configuration Tool also creates the `plugin-cfg.xml` file in the *plugins_root*/`config/`*`web_server_name`* directory.

    The web server reads the `plugin-cfg.xml` file to determine the applications that the application server on Machine A can serve to the web server on Machine B. Whenever the configuration changes, the application server regenerates the file. When regeneration occurs, propagate, or copy the actual `plugin-cfg.xml` file from the application server machine to the web server machine. You can automatically propagate the file to the IBM HTTP Server product.
18. Verify the success of the installation on the summary panel, and click **Finish**.

    If a problem occurs and the installation is unsuccessful, examine the logs in the *plugins_root*/`logs` directory. Correct any problems and re-configure.
19. Copy the `configure`*`web_server_name`* script from Machine B (the machine with the web server) to the *app_server_root* /`bin` directory on Machine A (the application server machine).

    web_server_name is the nickname of the web server that you specified. *web_server_name* is not a vendor name, such as IIS or Apache.

    On an operating system such as AIX or Linux, the file is `configure`*`web_server_name`*`.sh`. On a Windows system, the file is `configure`*`web_server_name`*`.bat`. For example, on a Linux system with an IBM HTTP Server named web_server_1 in the default location, copy *plugins_root*/`bin/configureweb_server_1.sh` from Machine B (the machine with the web server) to the *app_server_root*/`bin` directory on Machine A (the application server machine).

    If one platform is a system such as AIX or Linux and the other is a Windows platform, copy the script from the `crossPlatformScripts` directory. For example:
    - AIX HP-UX Linux Solaris *plugins_root*/`bin/configure`*`web_server_name`*`.sh`
    - Windows *plugins_root*/`bin/crossPlatformScripts/windows/configure`*`web_server_name`*`.bat`
20. Compensate for file encoding differences to prevent script failure.

The content of the `configure`*`web_server_name`*`.bat` script or the `configure`*`web_server_name`*`.sh` script can be corrupt if the default file encoding of the two machines differs. This scenario is possible when one machine is set up for a double-byte character set (DBCS) locale and the other machine is not.

Determine the file encoding and use one of the following procedures to circumvent the failure. To determine the default file encoding, run the appropriate command.

- Run the locale charmap command on a system such as AIX or Linux.
- Run the CHCP command on a Windows machine.

Use the result of the command on each machine as the value for the *web_server_machine_encoding* variable and the *application_server_machine_encoding* variable in one of the following procedures.

Procedures for compensating for encoding differences

- **Web server running on a system such as AIX or Linux**

  Suppose that the web server is running on a Linux machine and the application server is running on a Windows machine. Before you FTP the web server definition configuration script to the Windows machine in binary mode, run the following command on the system to encode the file:

```
iconv -f web_server_machine_encoding \
   -t application_server_machine_encoding \
   configureweb_server_name.bat
```

  Important: The name of the web server (nick name) is used in the name of the script file. The name cannot contain characters from a double-byte character set (DBCS) if you intend to set up IBM HTTP Server for automatic propagation.

- **Web server running on a Windows system**

  Suppose that the web server is running on a Windows machine and the application server is running on a machine with a system such as AIX or Linux. You must first download the iconv utility from a third party because the command is not included by default on Windows systems. Before you FTP the web server definition configuration script in binary mode to a system such as AIX or Linux, run the following command on the machine to encode the file:

```
iconv -f web_server_machine_encoding \
   -t application_server_machine_encoding \
   configureweb_server_name.sh
```

  For example, if the target machine is z/OS, you might use this command to convert the file from ASCII to EBCDIC, handling the end-of-line characters correctly:

```
iconv -f ISO8859-1 -t IBM-1047 configureweb_server_name.sh > new_script_name.sh
```

Omit the continuation characters (\) if you enter the command on one line.

If the conversion mapping is not supported by the iconv command on your system, copy the contents of the web server configuration script to a clip board and paste it onto the machine where the application server is running.

Note: If you copy over a `.sh` file onto a UNIX-based operating system after remote configuration on a Windows operating system, you must perform chmod 755.

21. Start the application server on Machine A.

    Use the startServer command, for example:

    - AIX HP-UX Linux Solaris *`profile_root`*`/bin/startServer.sh server1`
    - Windows *`profile_root`*`\bin\startServer server1`

22. Open a command window and change to the profile directory where the web server should be assigned. Run the script that you copied to Machine A (the application server machine). You need the following parameters:

    - Profile Name
    - (Optional) Admin user ID
    - (Optional) Admin user password

    For example, you could enter the following:

```
configurewebserver1.sh AppSrv01 my_user_ID my_Password
```

The web server will be configured via wsadmin.

The contents of the configurewebserver1.sh script will be similar to this:

```
wsadmin.bat -profileName AppSrv01 -user my_user_ID -password my_Password
   -f "%WAS_HOME%\bin\configureWebserverDefinition.jacl" webserver1 IHS..
```

23. From the administrative console of the deployment manager, click **System administration > Save Changes to Master Repository > Synchronize changes with Nodes > Save**.
24. **Domino Web Server only:** Set the WAS_PLUGIN_CONFIG_FILE environment variable.

    On platforms such as AIX or Linux, sourcing a script to the parent shell allows child processes to inherit the exported variables. On Windows systems, run the script as you would run any other command. Sourcing is automatic on Windows systems.

    a. Open a command window.
    b. Change directories to the plug-ins installation root directory.
    c. Issue the appropriate command for the *plugins_root*/`bin/setupPluginCfg.sh` script:
       - AIX HP-UX Solaris `. plugins_root/bin/setupPluginCfg.sh` (Notice the space between the period and the installation root directory.)
       - Linux `source plugins_root/bin/setupPluginCfg.sh`

    The script is also in the *lotus_root*/`notesdata` directory on operating systems such as AIX or Linux.

    Issue the appropriate command for the script before starting the Domino Web Server.
25. Regenerate the `plugin-cfg.xml` file on Machine A (the application server machine) using the administrative console. Click **Servers > Server Types > Web servers**. Select the web server, then click **Generate Plug-in**.

    During the installation of the plug-ins, the default `plugin-cfg.xml` file is installed on Machine B (the machine with the web server) in the *plugins_root*/`config/`*web_server_name* directory. The web server plug-in configuration service regenerates the `plugin-cfg.xml` file automatically. To use the current `plugin-cfg.xml` file from the application server, propagate the `plugin-cfg.xml` file as described in the next step.

    This step shows you how to regenerate the `plugin-cfg.xml` file. WebSphere Application Server products are configured to automatically regenerate the file each time a significant event occurs. Such events include installing applications on the application server and the web server, for example. Creating a new virtual host is another such event.
26. Propagate the `plugin-cfg.xml` file from the application server to the web server using the administrative console. Click **Servers > Web server**. Select the web server, then click **Propagate Plug-in**. Web servers other than IBM HTTP Server require manual propagation.

    The web server plug-in configuration service propagates the `plugin-cfg.xml` file automatically for IBM HTTP Server 8.0 only. For all other web servers, propagate the plug-in configuration file by manually copying the `plugin-cfg.xml` file from the *profile_root*/`config/cells/`*cell_name*/`nodes/`*node_name*/`servers/`*web_server_name* directory on Machine A (the application server machine) to the *plugins_root*/`config/`*web_server_name* directory on Machine B (the machine with the web server).
27. Start the Snoop servlet to verify the ability of the web server to retrieve an application from the application server.

    Test your environment by starting your application server, your web server, and using the Snoop servlet with an IP address.

    a. Start the application server. In a Network Deployment environment, the Snoop servlet is available in the cell only if you included the DefaultApplication when adding the application server to the cell. The -includeapps option for the addNode command migrates the DefaultApplication to the cell. If the application is not present, skip this step.

       Change directories to the *profile_root*/`bin` directory and run the startServer command:
       - AIX HP-UX Linux Solaris `./startServer.sh server1`
       - Windows `startServer server1`

b. Start the IBM HTTP Server or the web server that you are using.

   Use a command window to change the directory to the IBM HTTP Server installed image, or to the installed image of your web server. Issue the appropriate command to start the web server, such as these commands for IBM HTTP Server:

   To start the IBM HTTP Server from the command line:

   Access the apache and apachectl commands in the `IBMHttpServer/bin` directory.

   - AIX HP-UX Linux Solaris `./apachectl start`
   - Windows `apache`

c. Point your browser to `http://localhost:9080/snoop` to test the internal HTTP transport provided by the application server. Point your browser to `http://`*`Host_name_of_Web_server_machine`*`/snoop` to test the web server plug-in.

   The HTTP Transport port is 9080 by default and must be unique for every profile. The port is associated with a virtual host named default_host, which is configured to host the installed DefaultApplication. The Snoop servlet is part of the DefaultApplication. Change the port to match your actual HTTP Transport port.

d. Verify that Snoop is running.

   Either Web address should display the Snoop Servlet - Request/Client Information page.

e. **Remote IBM HTTP Server only:**

   Verify that the automatic propagation function can work on a remote IBM HTTP Server by using the following steps. This procedure is not necessary for local web servers.

   1) Create a user=adminUser, password=adminPassword in the *IHS_root* `/conf/admin.passwd` file. For example: `c:\ws\ihs80\bin\htpasswd -cb c:\ws\ihs80\conf\admin.passwd adminUser adminPassword`
   2) Use the administrative console of the deployment manager or the application server to enter the User ID and password information that you created for the administrative user of IBM HTTP Server. Go to **Servers > Web server > *web_server_definition* > Remote Web server administration**. Set the following values: admin Port=8008, User Id=adminUser, Password=adminPassword.
   3) Set the correct read/write permissions for the `httpd.conf` file and the plugin-cfg.xml file. See the *IHS_root* `/logs/admin_ERROR. LOG` file for more information.

   Automatic propagation of the plug-in configuration file requires the IBM HTTP administrative server to be up and running. If you are managing an IBM HTTP Server using the WebSphere Application Server administrative console, the following error might display:

```
"Could not connect to IHS Administration server error"
```

   Perform the following procedure to correct the error:

   1) Verify that the IBM HTTP Server administration server is running.
   2) Verify that the web server host name and the port that is defined in the WebSphere Application Server administrative console matches the IBM HTTP Server administration host name and port.
   3) Verify that the fire wall is not preventing you from accessing the IBM HTTP Server administration server from the WebSphere Application Server administrative console.
   4) Verify that the user ID and password that is specified in the WebSphere Application Server administrative console under remote managed, is created in the `admin.passwd` file, using the htpasswd command.
   5) If you are trying to connect securely, verify that you export the IBM HTTP Server administration server keydb personal certificate into the WebSphere Application Server key database as a signer certificate. This key database is specified by the com.ibm.ssl.trustStore directive in the `sas.client.props` file in the profile where your administrative console is running. This consideration is primarily for self-signed certificates.

6) If you still have problems, check the IBM HTTP Server `admin_ERROR. LOG` file and the WebSphere Application Server logs (`trace.log` file) to determine the cause of the problem.

Results

This procedure results in the installation of the Web Server Plug-ins for WebSphere Application Server on a web server machine. The Web Server Plug-ins Configuration Tool also configures the web server to support an application server on a separate machine.

The installation of the Web Server Plug-ins results in the creation of the `Plugins` directory and several subdirectories. The following directories are among those created on a Linux system, for example:
- *plugins_root*/`bin` contains the binary plug-ins for all supported web servers
- *plugins_root*/`logs` contains log files
- *plugins_root*/`properties` contains version information

What to do next

See "Selecting a web server topology diagram and roadmap" on page 61 for an overview of the installation procedure.

See "Web server configuration" on page 39 for more information about the files involved in configuring a web server.

See "Plug-ins configuration" on page 91 for information about the location of the plug-in configuration file.

See "Editing web server configuration files" on page 12 for information about how the Web Server Plug-ins Configuration Tool configures supported web servers.

Configuring multiple web servers and remote standalone application servers

This topic describes the procedure of installing and configuring multiple web servers and application servers on separate machines.

Before you begin

When multiple profiles exist, you can select the profile that the Web Server Plug-ins Configuration Tool configures. See "Plug-ins configuration" on page 91 for a description of the flow of logic that determines how to select the profile to configure.

If the WebSphere Application Server product family supports a particular brand of web server, such as IBM HTTP Server or Microsoft Internet Information Services (IIS), your WebSphere Application Server product provides a binary plug-in for the web server that you must install.

If the WebSphere Application Server product family does not provide a binary plug-in for a particular brand of web server, then the web server is not supported. The purpose of the binary plug-in is to provide the communication protocol between the web server and the application server.

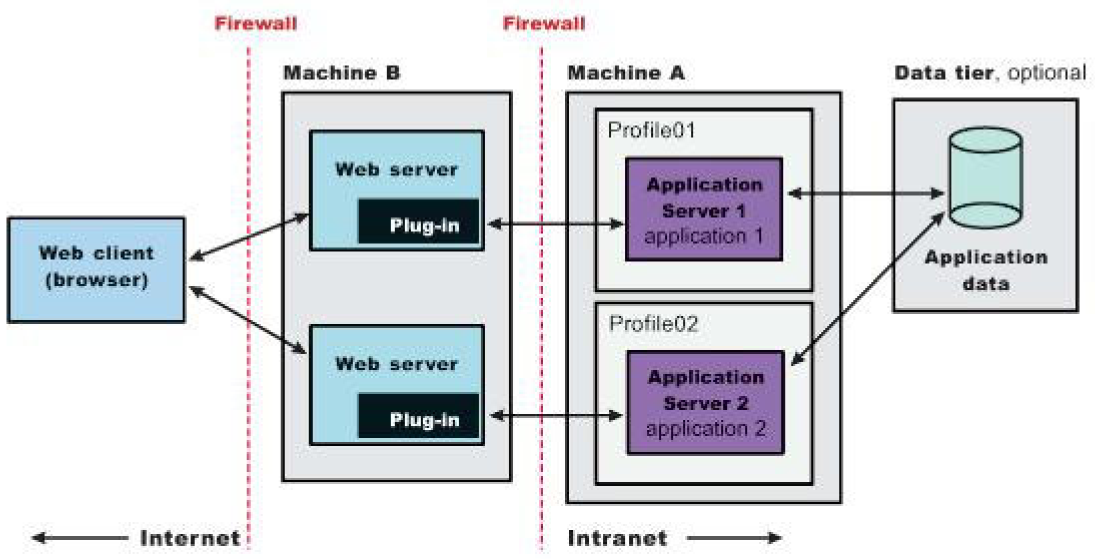
Suppose that you create a new profile and you also want to use a web server. You must install a new web server for the new profile, install the Web Server Plug-ins, and use the Web Server Plug-ins Configuration Tool to configure both the web server and the application server.

If the web server is not already installed, you can still install the Web Server Plug-ins for future use.

About this task

Installing the Web Server Plug-ins installs the plug-in module. The Web Server Plug-ins Configuration Tool configures the web server for communicating with the application server and creates a web server configuration definition in the application server, if possible.

This article describes how to create the following topology:



Perform the following procedure to install the plug-ins and configure both web servers and both application servers.

This topology lets each profile have unique applications, configuration settings, data, and log files, while sharing the same set of system files. Creating multiple profiles creates multiple application server environments that you can then dedicate to different purposes.

For example, each application server on a website can serve a different application. In another example, each application server can be a separate test environment that you assign to a programmer or a development team.

Tip: As an alternative to using the Web Server Plug-ins Configuration Tool, you can use the pct command-line tool with a response file to configure a web server. Read “Configuring a web server plug-in using the WCT command-line utility” on page 132 for more information.

Procedure

1. Install Installation Manager on Machine A and Machine B.
2. Use Installation Manager to install WebSphere Application Server Network Deployment on Machine A.
3. Create the first application server profile using the Profile Management Tool on Machine A.
4. Use Installation Manager to install the following on Machine B.
   - Web Server Plug-ins for WebSphere Application Server
   - Websphere Customization Toolbox
5. Use Installation Manager to install the IBM HTTP Server on Machine B, or install another supported web server on Machine B.
6. Open the WebSphere Customization Toolbox, and launch the Web Server Plug-ins Configuration Tool on the machine with the web server.

7. Select a web server plug-in runtime location.

   If the location of a previously installed web server that you want to use is not in the list, perform the following actions to add the location to your working set:

   a. Click **Add**.
   b. Enter a name for the web server plug-in location.
   c. Perform one of the following actions:
      - Enter the location.
      - Click **Browse**, find the location, and click **OK**.
8. Click **Create**.
9. Select the type of web server that you are configuring, and click **Next**.
10. Select the architecture of your installed target web server (64 bit or 32 bit) and click **Next** if you are asked.
11. Click **Browse** to select the configuration file or files for your web server, verify that the web server port is correct, and then click **Next** when you are finished.

    Select the file and not just the directory of the file. Some web servers have two configuration files and require you to browse for each file.

    The following list shows configuration files for supported web servers:

    Apache HTTP Server
    apache_root/`config/httpd.conf`

    Domino Web Server
    `names.nsf` and `Notes.jar`

    The wizard prompts for the `notes.jar` file. The actual name is `Notes.jar`.

    The Web Server Plug-ins Configuration Tool verifies that the files exist but the tool does not validate either file.

    IBM HTTP Server
    IHS_root/`conf/httpd.conf`

    Microsoft Internet Information Services (IIS)
    The Web Server Plug-ins Configuration Tool can determine the correct files to edit.

    Sun Java System Web Server (formerly Sun ONE Web Server and iPlanet Web Server) Version 6.0 and later
    `obj.conf` and `magnus.conf`
12. If you are configuring an IBM HTTP web server plug-in, perform the following actions.

    a. Optionally, set up the administration server configuration to administer the web server.
       1) Select **Setup IBM HTTP Server Administration Server**.
       2) Specify a port number on which the IBM HTTP administration server will communicate.
       3) Optionally, select **Create a user ID for IBM Server Administration Server authentication** and enter a user ID and password to authenticate to the IBM HTTP Server administrative server from the administrative console.
    b. Click **Next**.
    c. AIX HP-UX Linux Solaris Specify the system user ID and group to have write permission to IBM HTTP Server, the IBM HTTP Server administrative server, and the web server plug-in configuration files.

       Select **Create a new unique system user ID and group using the credentials** if necessary.
    d. Windows Optionally, set up the IBM HTTP Server Administration Server to run as a Window service.
       1) Select **Run IBM HTTP Server Administration Server as a Windows Service**.
       2) Perform one of the following actions:

        - Select **Log on as a local system account**.
        - Select **Log on as a specified user account**, and enter the user ID and password for that account.

          The user ID requires the following advanced user rights:

          – Act as part of the operating system

          – Log on as a service

    3) Choose whether your startup type will be automatic or manual.

  e. Click **Next**.

13. Specify a unique name for the web server definition, and click **Next**.
14. Select the configuration scenario.
    a. Choose the remote scenario.
    b. Identify the host name or IP address of Machine A, which is the application server machine.
    c. Click **Next**.
15. Select the profile to configure with the current web server plug-in, and click **Next**.
16. Examine the summary panel, and click **Configure** to begin configuring.

    The panel notifies you that you have manual steps to perform to complete the installation and configuration.

    The Web Server Plug-ins Configuration Tool creates the `configure`*`web_server_name`* script in the *plugins_root*/`bin/` directory on Machine B (the machine with the web server).

    The Web Server Plug-ins Configuration Tool also creates the `plugin-cfg.xml` file in the *plugins_root*/`config/`*`web_server_name`* directory.

    The web server reads the `plugin-cfg.xml` file to determine the applications that the application server on Machine A can serve to the web server on Machine B. Whenever the configuration changes, the application server regenerates the file. When regeneration occurs, propagate, or copy the actual `plugin-cfg.xml` file from the application server machine to the web server machine. You can automatically propagate the file to the IBM HTTP Server product.
17. Verify the success of the installation on the summary panel, and click **Finish**.

    If a problem occurs and the installation is unsuccessful, examine the logs in the *plugins_root*/`logs` directory. Correct any problems and re-configure.
18. Copy the `configure`*`web_server_name`* script from Machine B (the machine with the web server) to the *app_server_root* /`bin` directory on Machine A (the application server machine).

    web_server_name is the nickname of the web server that you specified. *web_server_name* is not a vendor name, such as IIS or Apache.

    On an operating system such as AIX or Linux, the file is `configure`*`web_server_name`*`.sh`. On a Windows system, the file is `configure`*`web_server_name`*`.bat`. For example, on a Linux system with an IBM HTTP Server named web_server_1 in the default location, copy *plugins_root*/`bin/configureweb_server_1.sh` from Machine B (the machine with the web server) to the *app_server_root*/`bin` directory on Machine A (the application server machine).

    If one platform is a system such as AIX or Linux and the other is a Windows platform, copy the script from the `crossPlatformScripts` directory. For example:
    - AIX HP-UX Linux Solaris *plugins_root*/`bin/configure`*`web_server_name`*`.sh`
    - Windows *plugins_root*/`bin/crossPlatformScripts/windows/configure`*`web_server_name`*`.bat`
19. Compensate for file encoding differences to prevent script failure.

    The content of the `configure`*`web_server_name`*`.bat` script or the `configure`*`web_server_name`*`.sh` script can be corrupt if the default file encoding of the two machines differs. This scenario is possible when one machine is set up for a double-byte character set (DBCS) locale and the other machine is not.

    Determine the file encoding and use one of the following procedures to circumvent the failure. To determine the default file encoding, run the appropriate command.
    - Run the locale charmap command on a system such as AIX or Linux.

- Run the CHCP command on a Windows machine.

Use the result of the command on each machine as the value for the *web_server_machine_encoding* variable and the *application_server_machine_encoding* variable in one of the following procedures.

Procedures for compensating for encoding differences

- **Web server running on a system such as AIX or Linux**

  Suppose that the web server is running on a Linux machine and the application server is running on a Windows machine. Before you FTP the web server definition configuration script to the Windows machine in binary mode, run the following command on the system to encode the file:

```
iconv -f web_server_machine_encoding \
   -t application_server_machine_encoding \
   configureweb_server_name.bat
```

  Important: The name of the web server (nick name) is used in the name of the script file. The name cannot contain characters from a double-byte character set (DBCS) if you intend to set up IBM HTTP Server for automatic propagation.

- **Web server running on a Windows system**

  Suppose that the web server is running on a Windows machine and the application server is running on a machine with a system such as AIX or Linux. You must first download the iconv utility from a third party because the command is not included by default on Windows systems. Before you FTP the web server definition configuration script in binary mode to a system such as AIX or Linux, run the following command on the machine to encode the file:

```
iconv -f web_server_machine_encoding \
   -t application_server_machine_encoding \
   configureweb_server_name.sh
```

  For example, if the target machine is z/OS, you might use this command to convert the file from ASCII to EBCDIC, handling the end-of-line characters correctly:

```
iconv -f ISO8859-1 -t IBM-1047 configureweb_server_name.sh > new_script_name.sh
```

Omit the continuation characters (\) if you enter the command on one line.

If the conversion mapping is not supported by the iconv command on your system, copy the contents of the web server configuration script to a clip board and paste it onto the machine where the application server is running.

Note: If you copy over a `.sh` file onto a UNIX-based operating system after remote configuration on a Windows operating system, you must perform chmod 755.

20. Start the application server on Machine A.

    Use the startServer command, for example:

    - AIX HP-UX Linux Solaris *profile_root*/bin/startServer.sh server1
    - Windows *profile_root*\bin\startServer server1

21. Open a command window and change to the profile directory where the web server should be assigned. Run the script that you copied to Machine A (the application server machine). You need the following parameters:

    - Profile Name
    - (Optional) Admin user ID
    - (Optional) Admin user password

    For example, you could enter the following:

```
configurewebserver1.sh AppSrv01 my_user_ID my_Password
```

    The web server will be configured via wsadmin.

    The contents of the configurewebserver1.sh script will be similar to this:

```
wsadmin.bat -profileName AppSrv01 -user my_user_ID -password my_Password
   -f "%WAS_HOME%\bin\configureWebserverDefinition.jacl" webserver1 IHS..
```

22. From the administrative console of the deployment manager, click **System administration > Save Changes to Master Repository > Synchronize changes with Nodes > Save**.
23. **Domino Web Server only:** Set the WAS_PLUGIN_CONFIG_FILE environment variable.

    On platforms such as AIX or Linux, sourcing a script to the parent shell allows child processes to inherit the exported variables. On Windows systems, run the script as you would run any other command. Sourcing is automatic on Windows systems.

    a. Open a command window.
    b. Change directories to the plug-ins installation root directory.
    c. Issue the appropriate command for the *plugins_root*/`bin/setupPluginCfg.sh` script:
       - AIX HP-UX Solaris `.` *`plugins_root`*`/bin/setupPluginCfg.sh` (Notice the space between the period and the installation root directory.)
       - Linux `source` *`plugins_root`*`/bin/setupPluginCfg.sh`

    The script is also in the *lotus_root*/`notesdata` directory on operating systems such as AIX or Linux.

    Issue the appropriate command for the script before starting the Domino Web Server.
24. Regenerate the `plugin-cfg.xml` file on Machine A (the application server machine) using the administrative console. Click **Servers > Server Types > Web servers**. Select the web server, then click **Generate Plug-in**.

    During the installation of the plug-ins, the default `plugin-cfg.xml` file is installed on Machine B (the machine with the web server) in the *plugins_root*/`config/`*`web_server_name`* directory. The web server plug-in configuration service regenerates the `plugin-cfg.xml` file automatically. To use the current `plugin-cfg.xml` file from the application server, propagate the `plugin-cfg.xml` file as described in the next step.

    This step shows you how to regenerate the `plugin-cfg.xml` file. WebSphere Application Server products are configured to automatically regenerate the file each time a significant event occurs. Such events include installing applications on the application server and the web server, for example. Creating a new virtual host is another such event.
25. Propagate the `plugin-cfg.xml` file from the application server to the web server using the administrative console. Click **Servers > Web server**. Select the web server, then click **Propagate Plug-in**. Web servers other than IBM HTTP Server require manual propagation.

    The web server plug-in configuration service propagates the `plugin-cfg.xml` file automatically for IBM HTTP Server 8.0 only. For all other web servers, propagate the plug-in configuration file by manually copying the `plugin-cfg.xml` file from the *profile_root*/`config/cells/`*cell_name*/`nodes/`*node_name*/`servers/`*web_server_name* directory on Machine A (the application server machine) to the *plugins_root*/`config/`*web_server_name* directory on Machine B (the machine with the web server).
26. Start the Snoop servlet to verify the ability of the web server to retrieve an application from the application server.

    Test your environment by starting your application server, your web server, and using the Snoop servlet with an IP address.

    a. Start the application server. In a Network Deployment environment, the Snoop servlet is available in the cell only if you included the DefaultApplication when adding the application server to the cell. The -includeapps option for the addNode command migrates the DefaultApplication to the cell. If the application is not present, skip this step.

       Change directories to the *profile_root*/`bin` directory and run the startServer command:
       - AIX HP-UX Linux Solaris `./startServer.sh server1`
       - Windows `startServer server1`

    b. Start the IBM HTTP Server or the web server that you are using.

       Use a command window to change the directory to the IBM HTTP Server installed image, or to the installed image of your web server. Issue the appropriate command to start the web server, such as these commands for IBM HTTP Server:

       To start the IBM HTTP Server from the command line:

Access the apache and apachectl commands in the `IBMHttpServer/bin` directory.

- AIX HP-UX Linux Solaris `./apachectl start`
- Windows `apache`

c. Point your browser to `http://localhost:9080/snoop` to test the internal HTTP transport provided by the application server. Point your browser to `http://`*`Host_name_of_Web_server_machine`*`/snoop` to test the web server plug-in.

   The HTTP Transport port is 9080 by default and must be unique for every profile. The port is associated with a virtual host named default_host, which is configured to host the installed DefaultApplication. The Snoop servlet is part of the DefaultApplication. Change the port to match your actual HTTP Transport port.

d. Verify that Snoop is running.

   Either Web address should display the Snoop Servlet - Request/Client Information page.

e. **Remote IBM HTTP Server only:**

   Verify that the automatic propagation function can work on a remote IBM HTTP Server by using the following steps. This procedure is not necessary for local web servers.

   1) Create a user=adminUser, password=adminPassword in the *IHS_root* `/conf/admin.passwd` file. For example: `c:\ws\ihs80\bin\htpasswd -cb c:\ws\ihs80\conf\admin.passwd adminUser adminPassword`
   2) Use the administrative console of the deployment manager or the application server to enter the User ID and password information that you created for the administrative user of IBM HTTP Server. Go to **Servers > Web server > *web_server_definition* > Remote Web server administration**. Set the following values: admin Port=8008, User Id=adminUser, Password=adminPassword.
   3) Set the correct read/write permissions for the `httpd.conf` file and the plugin-cfg.xml file. See the *IHS_root* `/logs/admin_ERROR. LOG` file for more information.

   Automatic propagation of the plug-in configuration file requires the IBM HTTP administrative server to be up and running. If you are managing an IBM HTTP Server using the WebSphere Application Server administrative console, the following error might display:

```
"Could not connect to IHS Administration server error"
```

   Perform the following procedure to correct the error:

   1) Verify that the IBM HTTP Server administration server is running.
   2) Verify that the web server host name and the port that is defined in the WebSphere Application Server administrative console matches the IBM HTTP Server administration host name and port.
   3) Verify that the fire wall is not preventing you from accessing the IBM HTTP Server administration server from the WebSphere Application Server administrative console.
   4) Verify that the user ID and password that is specified in the WebSphere Application Server administrative console under remote managed, is created in the `admin.passwd` file, using the htpasswd command.
   5) If you are trying to connect securely, verify that you export the IBM HTTP Server administration server keydb personal certificate into the WebSphere Application Server key database as a signer certificate. This key database is specified by the com.ibm.ssl.trustStore directive in the `sas.client.props` file in the profile where your administrative console is running. This consideration is primarily for self-signed certificates.
   6) If you still have problems, check the IBM HTTP Server `admin_ERROR. LOG` file and the WebSphere Application Server logs (`trace.log` file) to determine the cause of the problem.

27. Create the second application server profile using the Profile Management Tool on Machine A. Make the profile the default profile during the profile creation by selecting the check box on the appropriate panel.
28. Install a second IBM HTTP Server or another supported web server on Machine B.

29. On Machine B, configure the second web server using the Web Server Plug-ins Configuration Tool. Both web servers share a single installation of the plug-in binaries but must be configured individually.
30. The Installation Manager creates a script named `configure`*`web_server_name`* for the second web server. The script is in the *plugins_root*/`bin` directory on Machine B. Copy the script to the *app_server_root*/`bin` directory on Machine A.
31. Start the second application server.
32. Run the `configure`*`web_server_name`* script on Machine A to create a web server definition in the administrative console. You can then use the administrative console to manage the web server.
33. Propagate the `plugin-cfg.xml` file from the second application server to the web server using the administrative console. Click **Servers > Web server > Propagate Plug-in**. Web servers other than IBM HTTP Server require manual propagation.
34. Run the Snoop servlet on the second web server to verify that it is operational.

Results

This procedure results in installing two or more application servers on one machine and installing dedicated web servers on another machine. This procedure installs the Web Server Plug-ins for both web servers and configures both web servers and both application servers.

What to do next

See "Selecting a web server topology diagram and roadmap" on page 61 for an overview of the installation procedure.

See "Editing web server configuration files" on page 12 for information about how the Web Server Plug-ins Configuration Tool configures supported web servers.

See "Web server configuration" on page 39 for more information about the files involved in configuring a web server.

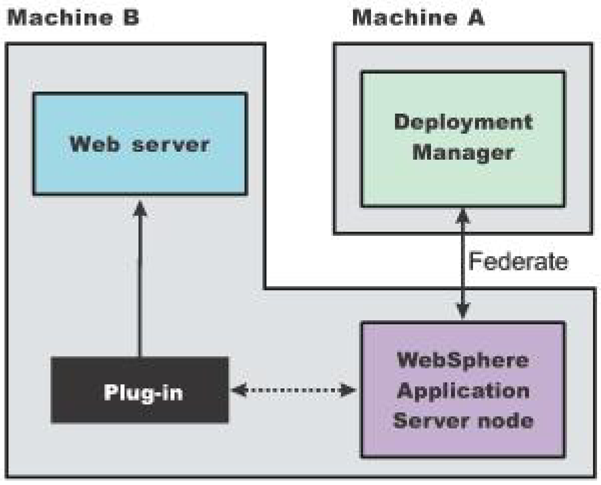
For IHS web servers, you can stop and start the web server and propagate the `plugin-cfg.xml` file from the WebSphere Application Server machine to the web server machine. For all other web servers, you cannot start/stop or propagate the `plugin-cfg.xml` file in the admin console. You will need to propagate the `plugin-cfg.xml` file manually. The following three steps describes how to perform manual propagation:
1. After completion of configuration with web servers other than IHS 6.x, verify that the `plugin-cfg.xml` file exists at `<WAS_HOME>/profiles/<PROFILE_HOME>/config/cells/<CELL_NAME>/nodes/<SERVER_NAME>/servers/<WEBSERVER_DEFINITION>`
2. Transfer the above `plugin-cfg.xml` to replace `<PLUGIN_HOME>/config/<WEBSERVER_DEFINITION>/plugin-xfg.xml`
3. Restart the web server and corresponding profile.

Configuring a web server and a custom profile on the same machine

This procedure describes installing a web server and its plug-in on a machine where the default profile is a custom profile.

Before you begin

When multiple profiles exist, you can select the profile that the Web Server Plug-ins Configuration Tool configures. See "Plug-ins configuration" on page 91 for a description of the flow of logic that determines how to select the profile to configure.



This procedure configures the custom profile on Machine B. This procedure assumes that you already have installed a deployment manager on Machine A.

The WebSphere Application Server node on Machine B is the custom node that you create in this procedure. This procedure starts the deployment manager and federates the custom node before installing the Web Server Plug-ins.

Start the deployment manager. The deployment manager must be running to successfully federate and configure the custom node.

About this task

Note: This topic references one or more of the application server log files. Beginning in WebSphere Application Server Version 8.0 you can configure the server to use the High Performance Extensible Logging (HPEL) log and trace infrastructure instead of using `SystemOut.log` , `SystemErr.log`, `trace.log`, and `activity.log` files or native z/OS logging facilities. If you are using HPEL, you can access all of your log and trace information using the LogViewer command-line tool from your server profile bin directory. See the information about using HPEL to troubleshoot applications for more information on using HPEL.

Use this procedure to install the web server plug-in, configure the web server, and create a web server definition in the custom profile (custom node).

Tip: As an alternative to using the Web Server Plug-ins Configuration Tool, you can use the pct command-line tool with a response file to configure a web server. Read “Configuring a web server plug-in using the WCT command-line utility” on page 132 for more information.

Procedure

1. Log on to the operating system.

   If you are installing as a nonroot or non-administrative user, then there are certain limitations.

   AIX HP-UX Linux Solaris In addition, select a umask that allows the owner to read/write to the files, and allows others to access them according to the prevailing system policy. For root, a umask of 022 is recommended. For nonroot users, a umask of 002 or 022 could be used, depending on whether or not the users share the group. To verify the umask setting, issue the following command:

```
umask
```

To set the umask setting to 022, issue the following command:

```
umask 022
```

Windows When installing as an administrative user on a Windows operating system, a Windows service is automatically created to autostart the application server. The installer user account must have the following advanced user rights:

- Act as part of the operating system
- Log on as a service

For example, on some Windows operating systems, click **Control Panel > Administrative Tools > Local Security Policy > Local Policies > User Rights Assignments** to set the advanced options. See your Windows operating system documentation for more information.

Windows If you plan to run the application server as a Windows service, do not install from a user ID that contains spaces. A user ID with spaces cannot be validated. Such a user ID is not allowed to continue the installation. To work around this restriction, install with a user ID that does not contain spaces.

2. Use Installation Manager to install the following on Machine B.
   - WebSphere Application Server Network Deployment
   - Web Server Plug-ins for WebSphere Application Server
   - Websphere Customization Toolbox
3. Use Installation Manager to install IBM HTTP Server on Machine B, or install another supported web server on Machine B.
4. Create a custom profile as the first profile Machine B, and federate the node as you create it.
5. Optional: Use the administrative console of the deployment manager to create an application server on the custom node.

   Click **Servers > Applications servers > New** and follow the instructions to create a server. A server is not required for installing the plug-ins but it lets you verify the functionality of the web server.
6. Optional: Install the DefaultApplication on the new server while you are in the administrative console of the deployment manager.

   The DefaultApplication includes the Snoop servlet. The verification step uses the Snoop servlet.
7. Open the WebSphere Customization Toolbox and launch the Web Server Plug-ins Configuration Tool on Machine B.
8. Select a web server plug-in runtime location.

   If the location of a previously installed web server plug-in that you want to use is not in the list, perform the following actions to add the location to your working set:
   a. Click **Add**.
   b. Enter a name for the web server plug-in location.
   c. Perform one of the following actions:
      - Enter the location.
      - Click **Browse**, find the location, and click **OK**.
9. Click **Create**.
10. Select the type of web server that you are configuring, and click **Next**.
11. Select the architecture of your installed target web server (64 bit or 32 bit) and click **Next** if you are asked.
12. Click **Browse** to select the configuration file or files for your web server, verify that the web server port is correct, and then click **Next** when you are finished.

    Select the file and not just the directory of the file. Some web servers have two configuration files and require you to browse for each file.

    The following list shows configuration files for supported web servers:

Apache HTTP Server
: *apache_root*/`config/httpd.conf`

Domino Web Server
: `names.nsf` and `Notes.jar`

  The wizard prompts for the `notes.jar` file. The actual name is `Notes.jar`.

  The Web Server Plug-ins Configuration Tool verifies that the files exist but the tool does not validate either file.

IBM HTTP Server
: *IHS_root*/`conf/httpd.conf`

Microsoft Internet Information Services (IIS)
: The Web Server Plug-ins Configuration Tool can determine the correct files to edit.

Sun Java System Web Server (formerly Sun ONE Web Server and iPlanet Web Server) Version 6.0 and later
: `obj.conf` and `magnus.conf`

13. If you are configuring an IBM HTTP web server plug-in, perform the following actions.
    a. Optionally, set up the administration server configuration to administer the web server.
       1) Select **Setup IBM HTTP Server Administration Server**.
       2) Specify a port number on which the IBM HTTP administration server will communicate.
       3) Optionally, select **Create a user ID for IBM Server Administration Server authentication** and enter a user ID and password to authenticate to the IBM HTTP Server administrative server from the administrative console.
    b. Click **Next**.
    c. AIX HP-UX Linux Solaris Specify the system user ID and group to have write permission to IBM HTTP Server, the IBM HTTP Server administrative server, and the web server plug-in configuration files.

       Select **Create a new unique system user ID and group using the credentials** if necessary.
    d. Windows Optionally, set up the IBM HTTP Server Administration Server to run as a Window service.
       1) Select **Run IBM HTTP Server Administration Server as a Windows Service**.
       2) Perform one of the following actions:
          - Select **Log on as a local system account**.
          - Select **Log on as a specified user account**, and enter the user ID and password for that account.

            The user ID requires the following advanced user rights:
            – Act as part of the operating system
            – Log on as a service
       3) Choose whether your startup type will be automatic or manual.
    e. Click **Next**.
14. Specify a unique name for the web server definition, and click **Next**.
15. Select the configuration scenario.
    a. Choose the local scenario.
    b. Perform one of the following actions:
       - Enter the installation location of WebSphere Application Server (*app_server_root*).
       - Click **Browse**, find the installation location of WebSphere Application Server (*app_server_root*), and click **OK**.
    c. Click **Next**.

16. Select the profile to configure with the current web server plug-in, and click **Next**.
17. Review the summary information, and click **Configure** to begin configuring the web server, web server plug-in, and profile.
18. Verify the success of the installation on the summary panel, and click **Finish**.

    If a problem occurs and the installation is unsuccessful, examine the logs in the *plugins_root*/`logs` directory. Correct any problems and re-configure.
19. Create the web server definition on Machine A.

    You can use the administrative console of the deployment manager to create the web server definition on a federated node; or you can run the configuration script that the Web Server Plug-ins Configuration Tool created.

    The script already contains all of the information that you must gather when using the administrative console option.
    - **Using the administrative console**

      Click **Servers > Web servers > New** and use the Create new web server entry wizard to create the web server definition.
    - **Running the configuration script**
      a. Paste the `configure`*`web_server_name`* script from Machine B to the *`app_server_root`*`/bin` directory on Machine A.
      b. Issue the appropriate command from a command window:
         - AIX HP-UX Linux Solaris `./`*`plugins_root`*`/bin/configure`*`web_server_name`*`.sh`
         - Windows *`plugins_root`*`\bin\configure`*`web_server_name`*`.bat`

      If you have enabled security or changed the default JMX connector type, edit the script and include the appropriate parameters on the wsadmin command.
20. **Domino Web Server only:** Set the WAS_PLUGIN_CONFIG_FILE environment variable.

    On platforms such as AIX or Linux, sourcing a script to the parent shell allows child processes to inherit the exported variables. On Windows systems, run the script as you would run any other command. Sourcing is automatic on Windows systems.
    a. Open a command window.
    b. Change directories to the plug-ins installation root directory.
    c. Issue the appropriate command for the *plugins_root*/`bin/setupPluginCfg.sh` script:
       - AIX HP-UX Solaris `. `*`plugins_root`*`/bin/setupPluginCfg.sh` (Notice the space between the period and the installation root directory.)
       - Linux `source `*`plugins_root`*`/bin/setupPluginCfg.sh`

    The script is also in the *lotus_root*/`notesdata` directory on operating systems such as AIX or Linux.

    Issue the appropriate command for the script before starting the Domino Web Server.
21. Start the Snoop servlet to verify the ability of the web server to retrieve an application from the application server.

    Test your environment by starting your application server, your web server, and using the Snoop servlet with an IP address.
    a. Start the application server. In a Network Deployment environment, the Snoop servlet is available in the cell only if you included the DefaultApplication when adding the application server to the cell. The -includeapps option for the addNode command migrates the DefaultApplication to the cell. If the application is not present, skip this step.

       Change directories to the *profile_root*/`bin` directory and run the startServer command:
       - AIX HP-UX Linux Solaris `./startServer.sh server1`
       - Windows `startServer server1`
    b. Start the IBM HTTP Server or the web server that you are using.

Use a command window to change the directory to the IBM HTTP Server installed image, or to the installed image of your web server. Issue the appropriate command to start the web server, such as these commands for IBM HTTP Server:

To start the IBM HTTP Server from the command line:

Access the apache and apachectl commands in the `IBMHttpServer/bin` directory.

- AIX HP-UX Linux Solaris `./apachectl start`
- Windows `apache`

c. Point your browser to `http://localhost:9080/snoop` to test the internal HTTP transport provided by the application server. Point your browser to `http://`*`Host_name_of_Web_server_machine`*`/snoop` to test the web server plug-in.

   The HTTP Transport port is 9080 by default and must be unique for every profile. The port is associated with a virtual host named default_host, which is configured to host the installed DefaultApplication. The Snoop servlet is part of the DefaultApplication. Change the port to match your actual HTTP Transport port.

d. Verify that Snoop is running.

   Either Web address should display the Snoop Servlet - Request/Client Information page.

e. **Remote IBM HTTP Server only:**

   Verify that the automatic propagation function can work on a remote IBM HTTP Server by using the following steps. This procedure is not necessary for local web servers.

   1) Create a user=adminUser, password=adminPassword in the *IHS_root* `/conf/admin.passwd` file. For example: `c:\ws\ihs80\bin\htpasswd -cb c:\ws\ihs80\conf\admin.passwd adminUser adminPassword`
   2) Use the administrative console of the deployment manager or the application server to enter the User ID and password information that you created for the administrative user of IBM HTTP Server. Go to **Servers > Web server >** ***web_server_definition*** **> Remote Web server administration**. Set the following values: admin Port=8008, User Id=adminUser, Password=adminPassword.
   3) Set the correct read/write permissions for the `httpd.conf` file and the plugin-cfg.xml file. See the *IHS_root* `/logs/admin_ERROR. LOG` file for more information.

   Automatic propagation of the plug-in configuration file requires the IBM HTTP administrative server to be up and running. If you are managing an IBM HTTP Server using the WebSphere Application Server administrative console, the following error might display:

```
"Could not connect to IHS Administration server error"
```

   Perform the following procedure to correct the error:

   1) Verify that the IBM HTTP Server administration server is running.
   2) Verify that the web server host name and the port that is defined in the WebSphere Application Server administrative console matches the IBM HTTP Server administration host name and port.
   3) Verify that the fire wall is not preventing you from accessing the IBM HTTP Server administration server from the WebSphere Application Server administrative console.
   4) Verify that the user ID and password that is specified in the WebSphere Application Server administrative console under remote managed, is created in the `admin.passwd` file, using the htpasswd command.
   5) If you are trying to connect securely, verify that you export the IBM HTTP Server administration server keydb personal certificate into the WebSphere Application Server key database as a signer certificate. This key database is specified by the com.ibm.ssl.trustStore directive in the `sas.client.props` file in the profile where your administrative console is running. This consideration is primarily for self-signed certificates.
   6) If you still have problems, check the IBM HTTP Server `admin_ERROR. LOG` file and the WebSphere Application Server logs (`trace.log` file) to determine the cause of the problem.

22. If the deployment manager does not have the DefaultApplication installed, you can test the functionality of the web server and the custom node using an application of your own.
23. From the administrative console of the deployment manager, click **System administration** > **Save Changes to Master Repository** > **Synchronize changes with Nodes** > **Save**.
24. To create multiple web server definitions for the managed node, use the Web Server Plug-ins Configuration Tool to configure each web server.

    Identify the same managed node each time. Give each web server a different nick name.

Results

This procedure results in the installation of the Web Server Plug-ins for WebSphere Application Server on a web server machine. The Web Server Plug-ins Configuration Tool creates a web server definition within the managed node.

The Web Server Plug-ins Configuration Tool configures the web server to use the `plugin-cfg.xml` file that is within the managed custom node.

The deployment manager regenerates the web server plug-in configuration file, `plugin-cfg.xml` whenever an event occurs that affects the file. Such events include the addition or removal of an application, server, or virtual host.

The creation or removal of clusters and cluster members also causes file regeneration. Automatic propagation through node synchronization copies the file after each regeneration to the following location on the custom node machine:

```
profile_root
   /config/cells/cell_name/nodes/
   node_name_of_custom_profile/servers/
   web_server_name/plugin-cfg.xml
```

The installation of the Web Server Plug-ins results in the creation of the `Plugins` directory and several subdirectories. The following directories are among those created on a Linux system, for example:

- *plugins_root*/`bin` contains the binary plug-ins for all supported web servers
- *plugins_root*/`logs` contains log files
- *plugins_root*/`properties` contains version information

What to do next

See "Plug-ins configuration" on page 91 for information about the location of the plug-in configuration file.

See "Web server configuration" on page 39 for more information about the files involved in configuring a web server.

See "Editing web server configuration files" on page 12 for information about how the Web Server Plug-ins Configuration Tool configures supported web servers.

See "Installing and configuring web server plug-ins" on page 58 for information about other installation scenarios for installing Web server plug-ins.

Configuring a web server and a deployment manager profile on the same machine

This procedure describes installing a web server and its plug-in on a machine that contains a deployment manager and a managed node.

Before you begin

When multiple profiles exist, you can select the profile that the Web Server Plug-ins Configuration Tool configures. See “Plug-ins configuration” on page 91 for a description of the flow of logic that determines how to select the profile to configure.

This procedure configures the deployment manager profile on the machine. A managed node must exist to define a web server definition, which is always on a managed node.

Start the deployment manager and the node agent for the managed node. The deployment manager and the node must be running to successfully change its configuration.

About this task

Tip: As an alternative to using the Web Server Plug-ins Configuration Tool, you can use the pct command-line tool with a response file to configure a web server. Read “Configuring a web server plug-in using the WCT command-line utility” on page 132 for more information.

Use this procedure to install the web server plug-in, configure the web server, and create a web server definition.

Procedure

1. Log on to the operating system.

   If you are installing as a nonroot or non-administrative user, then there are certain limitations.

   AIX HP-UX Linux Solaris In addition, select a umask that allows the owner to read/write to the files, and allows others to access them according to the prevailing system policy. For root, a umask of 022 is recommended. For nonroot users, a umask of 002 or 022 could be used, depending on whether or not the users share the group. To verify the umask setting, issue the following command:

```
umask
```

   To set the umask setting to 022, issue the following command:

```
umask 022
```

   Windows When installing as an administrative user on a Windows operating system, a Windows service is automatically created to autostart the application server. The installer user account must have the following advanced user rights:

   - Act as part of the operating system
   - Log on as a service

   For example, on some Windows operating systems, click **Control Panel > Administrative Tools > Local Security Policy > Local Policies > User Rights Assignments** to set the advanced options. See your Windows operating system documentation for more information.

   Windows If you plan to run the application server as a Windows service, do not install from a user ID that contains spaces. A user ID with spaces cannot be validated. Such a user ID is not allowed to continue the installation. To work around this restriction, install with a user ID that does not contain spaces.
2. Install IBM Installation Manager.
3. Use Installation Manager to install the WebSphere Application Server Network Deployment product.
4. Create a deployment manager profile as the first profile on the machine.
5. Create a standalone application server or custom profile, and federate the node.
6. Use Installation Manager to install the following:
   - Web Server Plug-ins for WebSphere Application Server
   - Websphere Customization Toolbox

7. Use Installation Manager to install IBM HTTP Server, or install another supported web server.
8. Open the WebSphere Customization Toolbox and launch the Web Server Plug-ins Configuration Tool.
9. Select a web server plug-in runtime location.

   If the location of a previously installed web server plug-in that you want to use is not in the list, perform the following actions to add the location to your working set:

   a. Click **Add**.
   b. Enter a name for the web server plug-in location.
   c. Perform one of the following actions:
      - Enter the location.
      - Click **Browse**, find the location, and click **OK**.
10. Click **Create**.
11. Select the type of web server that you are configuring, and click **Next**.
12. Select the architecture of your installed target web server (64 bit or 32 bit) and click **Next** if you are asked.
13. Click **Browse** to select the configuration file or files for your web server, verify that the web server port is correct, and then click **Next** when you are finished.

    Select the file and not just the directory of the file. Some web servers have two configuration files and require you to browse for each file.

    The following list shows configuration files for supported web servers:

    Apache HTTP Server
    : *apache_root*/`config/httpd.conf`

    Domino Web Server
    : `names.nsf` and `Notes.jar`

      The wizard prompts for the `notes.jar` file. The actual name is `Notes.jar`.

      The Web Server Plug-ins Configuration Tool verifies that the files exist but the tool does not validate either file.

    IBM HTTP Server
    : *IHS_root*/`conf/httpd.conf`

    Microsoft Internet Information Services (IIS)
    : The Web Server Plug-ins Configuration Tool can determine the correct files to edit.

    Sun Java System Web Server (formerly Sun ONE Web Server and iPlanet Web Server) Version 6.0 and later
    : `obj.conf` and `magnus.conf`
14. If you are configuring an IBM HTTP web server plug-in, perform the following actions.
    a. Optionally, set up the administration server configuration to administer the web server.
       1) Select **Setup IBM HTTP Server Administration Server**.
       2) Specify a port number on which the IBM HTTP administration server will communicate.
       3) Optionally, select **Create a user ID for IBM Server Administration Server authentication** and enter a user ID and password to authenticate to the IBM HTTP Server administrative server from the administrative console.
    b. Click **Next**.
    c. AIX HP-UX Linux Solaris Specify the system user ID and group to have write permission to IBM HTTP Server, the IBM HTTP Server administrative server, and the web server plug-in configuration files.

       Select **Create a new unique system user ID and group using the credentials** if necessary.
    d. Windows Optionally, set up the IBM HTTP Server Administration Server to run as a Window service.

            1) Select **Run IBM HTTP Server Administration Server as a Windows Service**.
            2) Perform one of the following actions:
                - Select **Log on as a local system account**.
                - Select **Log on as a specified user account**, and enter the user ID and password for that account.

                  The user ID requires the following advanced user rights:
                  - Act as part of the operating system
                  - Log on as a service
            3) Choose whether your startup type will be automatic or manual.
        e. Click **Next**.
15. Specify a unique name for the web server definition, and click **Next**.
16. Select the configuration scenario.
    a. Choose the local scenario.
    b. Perform one of the following actions:
        - Enter the installation location of WebSphere Application Server (*app_server_root*).
        - Click **Browse**, find the installation location of WebSphere Application Server (*app_server_root*), and click **OK**.
    c. Click **Next**.
17. Select the profile to configure with the current web server plug-in, and click **Next**.
18. Review the summary information, and click **Configure** to begin configuring the web server, web server plug-in, and profile.
19. Verify the success of the installation on the summary panel, and click **Finish**.

    If a problem occurs and the installation is unsuccessful, examine the logs in the *plugins_root*/`logs` directory. Correct any problems and re-configure.
20. **Domino Web Server only:** Set the WAS_PLUGIN_CONFIG_FILE environment variable.

    On platforms such as AIX or Linux, sourcing a script to the parent shell allows child processes to inherit the exported variables. On Windows systems, run the script as you would run any other command. Sourcing is automatic on Windows systems.
    a. Open a command window.
    b. Change directories to the plug-ins installation root directory.
    c. Issue the appropriate command for the *plugins_root*/`bin/setupPluginCfg.sh` script:
        - AIX HP-UX Solaris `. `*`plugins_root`*`/bin/setupPluginCfg.sh` (Notice the space between the period and the installation root directory.)
        - Linux `source `*`plugins_root`*`/bin/setupPluginCfg.sh`

    The script is also in the *lotus_root*/`notesdata` directory on operating systems such as AIX or Linux.

    Issue the appropriate command for the script before starting the Domino Web Server.
21. From the administrative console of the deployment manager, click **System administration > Save Changes to Master Repository > Synchronize changes with Nodes > Save**.
22. Start the Snoop servlet to verify the ability of the web server to retrieve an application from the application server.

    Test your environment by starting your application server, your web server, and using the Snoop servlet with an IP address.
    a. Start the application server. In a Network Deployment environment, the Snoop servlet is available in the cell only if you included the DefaultApplication when adding the application server to the cell. The -includeapps option for the addNode command migrates the DefaultApplication to the cell. If the application is not present, skip this step.

       Change directories to the *profile_root*/`bin` directory and run the startServer command:

- AIX HP-UX Linux Solaris `./startServer.sh server1`
- Windows `startServer server1`

b. Start the IBM HTTP Server or the web server that you are using.

   Use a command window to change the directory to the IBM HTTP Server installed image, or to the installed image of your web server. Issue the appropriate command to start the web server, such as these commands for IBM HTTP Server:

   To start the IBM HTTP Server from the command line:

   Access the apache and apachectl commands in the `IBMHttpServer/bin` directory.

   - AIX HP-UX Linux Solaris `./apachectl start`
   - Windows `apache`

c. Point your browser to `http://localhost:9080/snoop` to test the internal HTTP transport provided by the application server. Point your browser to `http://`*`Host_name_of_Web_server_machine`*`/snoop` to test the web server plug-in.

   The HTTP Transport port is 9080 by default and must be unique for every profile. The port is associated with a virtual host named default_host, which is configured to host the installed DefaultApplication. The Snoop servlet is part of the DefaultApplication. Change the port to match your actual HTTP Transport port.

d. Verify that Snoop is running.

   Either Web address should display the Snoop Servlet - Request/Client Information page.

Results

The installation of the Web Server Plug-ins results in the creation of the `Plugins` directory and several subdirectories. The following directories are among those created on a Linux system, for example:

- *plugins_root*/`bin` contains the binary plug-ins for all supported web servers
- *plugins_root*/`logs` contains log files
- *plugins_root*/`properties` contains version information

The Installation Manager configures the web server to use the *plugins_root*/`plugin-cfg.xml` file.

What to do next

After installing the binary plug-in for the local Web server, you must create a managed node before you can successfully run the configuration script and use the web server.

See “Plug-ins configuration” on page 91 for an overview of the installation procedure.

See “Web server configuration” on page 39 for more information about the files involved in configuring a web server.

See “Editing web server configuration files” on page 12 for information about how the Web Server Plug-ins Configuration Tool configures supported web servers.

See “Installing and configuring web server plug-ins” on page 58 for information about other installation scenarios for installing Web server plug-ins.

Configuring a web server plug-in using the WCT command-line utility

The WCT command-line utility invokes the command-line tool that is specified by the -tool parameter. You can use the WCT command-line utility and specify the pct tool to configure a web server to use an application server as a hosting server.

Procedure

Configure a web server to use an application server as a hosting server.

Location of the WCT command-line utility

The product includes the following script that sets up the environment and invokes the WCT command-line utility.

- Windows *WCT_install_root*\WCT\wctcmd.bat
- Linux *WCT_install_root*/WCT/wctcmd.sh

Syntax of the WCT command-line utility

Windows

```
wctcmd.bat
      -tool tool_ID
      -defLocPathname definition_location_pathname
      -createDefinitionLocation definition_location_name
      -importDefinitionLocation definition_location_name
      -removeDefinitionLocation definition_location_name
      -defLocName definition_location_name
      -defLocVersion definition_location_version
      -response response_file
      -listDefinitionLocations
      -deleteDefinition definition_name
      -listDefinitions
```

Linux

```
./wctcmd.sh
      -tool tool_ID
      -defLocPathname definition_location_pathname
      -createDefinitionLocation definition_location_name
      -importDefinitionLocation definition_location_name
      -removeDefinitionLocation definition_location_name
      -defLocName definition_location_name
      -defLocVersion definition_location_version
      -response response_file
      -listDefinitionLocations
      -deleteDefinition definition_name
      -listDefinitions
```

Parameters of the WCT command-line utility

-tool *tool_ID*

Specifies the name of the tool to launch as it is registered with the WCT command-line utility

This parameter is required.

-defLocPathname *definition_location_pathname*

Specifies the absolute path name of the definition location to use when the specified tool is launched

This parameter is required.

-createDefinitionLocation *definition_location_name*

Specifies that the WCT command-line utility should create a definition location

This parameter is optional.

-importDefinitionLocation *definition_location_name*

Specifies that the WCT command-line utility should import a definition location

This parameter is optional.

-removeDefinitionLocation *definition_location_name*

Specifies that the WCT command-line utility should remove a definition location

This parameter is optional.

-defLocName *definition_location_name*

Specifies the name of the definition location as it resides in the definition location registry

-defLocVersion *definition_location_version*
: Specifies the version of definition location to create

  This parameter is optional.

-response *response_file*
: Specifies the response file containing tool arguments

  This parameter is optional.

-listDefinitionLocations
: Lists the available definition locations.

-deleteDefinition *definition_name*
: Specifies that the WCT command-line utility should delete a definition

  This parameter is optional.

  The *definition_name* is required. Either one of the following parameters is also required:

  - `-defLocName` *definition_location_name*
  - `-defLocpathname` *definition_location_pathname*

  If both parameter values are supplied, the first one is used. If the first value supplied does not pass the validation check, the command fails with an error message.

-listDefinitions
: Lists the available definitions at a specified definition location or definition location path name

  Either one of the following parameters is required:

  - `-defLocName` *definition_location_name*
  - `-defLocpathname` *definition_location_pathname*

  If both parameter values are supplied, the first one is used. If the first value supplied does not pass the validation check, the command fails with an error message.

Notes:

- Command-line arguments are case sensitive.
- If an argument accepts a value containing spaces, the value must be enclosed in double quotes (" ").

Examples

Using the pct tool to configure an IHS Web Server to use an application server as a hosting server:

- Windows `wctcmd.bat -tool pct -defLocPathname C:\data\IBM\WebSphere\Plugins -defLocName someDefLocName -response C:\IBM\WebSphere\tools\WCT\responsefile.txt`
- Linux `./wctcmd.sh -tool pct -defLocPathname /data/IBM/WebSphere/Plugins -defLocName someDefLocName -response /var/IBM/WebSphere/tools/WCT/responsefile.txt`

The following is an example of the content of a response file for an IHS local plug-in configuration.

AIX HP-UX Linux Solaris

```
configType=local_standalone
enableAdminServerSupport=true
enableUserAndPass=true
enableWinService=false
ihsAdminCreateUserAndGroup=true
ihsAdminPassword=******
```

```
ihsAdminPort=8008
ihsAdminUnixUserGroup=grp101
ihsAdminUnixUserID=user1
mapWebServerToApplications=true
profileName=AppSrv01
wasExistingLocation=opt/IBM/WebSphere/AppServer80
webServerConfigFile1=opt/IBM/HTTPServer/conf/httpd.conf
webServerDefinition=webserver1
webServerHostName=local.ibm.com
webServerInstallArch=32
webServerOS=windows
webServerPortNumber=80
webServerSelected=ihs
```

Windows

```
configType=local_standalone
enableAdminServerSupport=true
enableUserAndPass=true
enableWinService=true
ihsAdminPassword=******
ihsAdminPort=8008
ihsAdminUserID=admin1
ihsWindowsPassword=******
ihsWindowsStartupType=Automatic
ihsWindowsUserID=user1
mapWebServerToApplications=true
profileName=AppSrv01
wasExistingLocation=D:\IBM\WebSphere\AppServer80
webServerConfigFile1=D:\IBM\HTTPServer\conf\httpd.conf
webServerDefinition=webserver1
webServerHostName=local.ibm.com
webServerInstallArch=32
webServerOS=windows
webServerPortNumber=80
webServerSelected=ihs
```

The following is an example of the content of a response file for an IHS remote plug-in configuration.

AIX HP-UX Linux Solaris

```
configType=remote
enableAdminServerSupport=true
enableUserAndPass=true
enableWinService=false
ihsAdminCreateUserAndGroup=true
ihsAdminPassword=******
ihsAdminPort=8008
ihsAdminUnixUserGroup=grp101
ihsAdminUnixUserID=user1
mapWebServerToApplications=true
profileName=AppSrv01
wasExistingLocation=opt/IBM/WebSphere/AppServer80
wasMachineHostname=192.168.1.2
webServerConfigFile1=opt/IBM/HTTPServer/conf/httpd.conf
webServerDefinition=webserver1
webServerHostName=remote.ibm.com
webServerInstallArch=32
webServerOS=windows
webServerPortNumber=80
webServerSelected=ihs
```

Windows

```
configType=remote
enableAdminServerSupport=true
enableUserAndPass=true
enableWinService=true
ihsAdminPassword=******
ihsAdminPort=8008
ihsAdminUserID=admin1
ihsWindowsPassword=******
ihsWindowsStartupType=Automatic
ihsWindowsUserID=user1
mapWebServerToApplications=true
profileName=AppSrv01
wasExistingLocation=D:\IBM\WebSphere\AppServer80
wasMachineHostname=192.168.1.2
webServerConfigFile1=D:\IBM\HTTPServer\conf\httpd.conf
webServerDefinition=webserver1
webServerHostName=remote.ibm.com
```

```
webServerInstallArch=32
webServerOS=windows
webServerPortNumber=80
webServerSelected=ihs
```

Importing a definition location for the pct tool:

Windows

```
wctcmd.bat -tool pct -importDefinitionLocation -defLocName someDefLocName -defLocPathname C:\data\IBM\WebSphere\Plugins -re
```

Linux

```
./wctcmd.sh -tool pct -importDefinitionLocation -defLocName someDefLocName -defLocPathname /data/IBM/WebSphere/Plugins -res
```

Removing a definition location for the pct tool:

Windows

```
wctcmd.bat -tool pct -removeDefinitionLocation -defLocName someDefLocName -defLocPathname C:\data\IBM\WebSphere\Plugins
```

Linux

```
./wctcmd.sh -tool pct -removeDefinitionLocation -defLocName someDefLocName -defLocPathname /data/IBM/WebSphere/Plugins
```

Listing the available definition locations for the pct tool:

Windows

```
wctcmd.bat -tool pct -defLocPathname C:\data\IBM\WebSphere\Plugins -listDefinitionLocations
```

Linux

```
./wctcmd.sh -tool pct -defLocPathname /data/IBM/WebSphere/Plugins -listDefinitionLocations
```

Removing a definition for the pct tool:

Windows

```
wctcmd.bat -tool pct -deleteDefinition someDefName -defLocName someDefLocName
```

Linux

```
./wctcmd.sh -tool pct -deleteDefinition someDefName -defLocName someDefLocName
```

Listing the available definitions for the pct tool:

Windows

```
wctcmd.bat -tool pct -defLocPathname C:\data\IBM\WebSphere\Plugins -listDefinitions
```

Linux

```
./wctcmd.sh -tool pct -defLocPathname /data/IBM/WebSphere/Plugins -listDefinition
```

Creating or updating a global web server plug-in configuration file

If all of the application servers in a cell use the same web server to route requests for dynamic content, such as servlets, from web applications to application servers, you can create a global web server plug-in configuration file for that cell. The resulting `plugin-cfg.xml` file is located in the `profile_root/config/cells` directory.

Before you begin

Note: You can update the *global* `plugin-cfg.xml` file using the administrative console or running the GenPluginCfg command for all of the clusters in a cell. However, you must delete the `config/cells/plugin-cfg.xml` file before you update the *global* `plugin-cfg.xml` file. If you do not delete the `config/cells/plugin-cfg.xml` file, only the new properties and their values are added to the *global* `plugin-cfg.xml` file. Any updates to existing plug-in property values are not added to the *global* `plugin-cfg.xml` file.

About this task

You must update the global web server plug-in configuration file whenever you perform one of the following actions:

- Change the configuration settings for an application server, cluster, virtual host, or web container transport that is part of that cell.
- Add a new application server, cluster, virtual host, or web container transport to that cell.

To update the configuration settings for a global web server plug-in, you can either use the Update global web server plug-in configuration page in the administrative console, or issue the following command:

```
%was_profile_home%/config/cells/GenPluginCfg.sh|bat
```

Both methods for regenerating the global web server plug-in configuration create a `plugin-cfg.xml` file in ASCII format.

To use the Update global web server plug-in configuration page in the administrative console:

Procedure

1. Click **Environment > Update global web server plug-in configuration**.
2. Click **OK** to update the `plugin-cfg.xml` file.
3. Optional: Click **View or download the current web server plug-in configuration file** if you want to view or download the current version of this file.

   You can select this option if you want to:

   - View the current version of the file before you update it.
   - View the file after it is updated.
   - Download a copy of this file to a remote machine.

Results

Regenerating the configuration might take a while to complete. After it finishes, all objects in the administrative cell use their newest settings, which the web server can access. Whether triggered manually or occurring automatically, plug-in regeneration requires about 30 to 60 seconds to complete when the application server is on the same physical machine (node) as the web server. In other cases, it takes more time.

The delay is important because it determines how soon the new plug-in configuration takes effect. Suppose you add a new served path for a servlet, then regenerate the plug-in configurations. The regeneration requires 40 seconds, after which a user should be able to access the servlet by the new served path.

For an HTTP plug-in, the length of the delay is determined by the Refresh Interval attribute of the Config element in the `plugin-cfg.xml` file. The plug-in polls the disk, or file system, at this interval to see whether the configuration has changed. The default interval is 60 seconds. To regenerate the plug-in configuration requires twice the refresh interval.

In a development environment in which you are frequently changing settings in the administrative console, it is recommended that you set the refresh interval to 3 to 5 seconds.

In a production environment, set a longer refresh interval, perhaps as long as 30 minutes, depending on the frequency of changes.

What to do next

You might need to stop the application servers in the cell and then start the application servers again before the changes to the plug-in configuration go into effect.

If the web server is running on a remote machine, click **View or download the current web server plug-in configuration file** to download a copy of the `plugin-cfg.xml` file to a that machine.

When the deployment manager is installed on a machine that is remote from where the product is installed, one of the following solutions must be implemented in order for the `plugin-cfg.xml` file to retain the application server directory structures, and not assume those of the deployment manager after the plug-in is regenerated and a full synchronization occurs.

- **Command line**:

  At a command prompt, enter the following command to change the `DeploymentManager/bin` directory and type on the machine where the deployment manager is installed. This command creates or updates the `plugin-cfg.xml` file, and changes all of the directories in the `plugin-cfg.xml` file to *app_server_root* directories.

  ```
  GenPluginCfg -destination.root <app_server_root>
  ```

  For example, issue the following command from the `DeploymentManager/bin` directory.

  ```
  GenPluginCfg -destination.root "E:\WebSphere\AppServer"
  ```

- **plugin-cfg.xml file**:

  Edit the `plugin-cfg.xml` file to point to the correct directory structure for the log file, keyring, and stashfile.

  Perform a full synchronization so the `plugin-cfg.xml` file is replicated in all the nodes. You can use scripting or the administrative console to synchronize the nodes in the cell.

  The deployment manager `plugin-cfg.xml` file can point to the application server directories without any conflict.

Creating or updating a global web server plug-in configuration file

If all of the application servers in a cell use the same web server to route requests for dynamic content, such as servlets, from web applications to application servers, you can create a global web server plug-in configuration file for that cell. The resulting `plugin-cfg.xml` file is located in the `profile_root/config/cells` directory.

Before you begin

Note: You can update the *global* `plugin-cfg.xml` file using the administrative console or running the GenPluginCfg command for all of the clusters in a cell. However, you must delete the `config/cells/plugin-cfg.xml` file before you update the *global* `plugin-cfg.xml` file. If you do not delete the `config/cells/plugin-cfg.xml` file, only the new properties and their values are added to the *global* `plugin-cfg.xml` file. Any updates to existing plug-in property values are not added to the *global* `plugin-cfg.xml` file.

About this task

You must update the global web server plug-in configuration file whenever you perform one of the following actions:

- Change the configuration settings for an application server, cluster, virtual host, or web container transport that is part of that cell.
- Add a new application server, cluster, virtual host, or web container transport to that cell.

To update the configuration settings for a global web server plug-in, you can either use the Update global web server plug-in configuration page in the administrative console, or issue the following command:

```
%was_profile_home%/config/cells/GenPluginCfg.sh|bat
```

Both methods for regenerating the global web server plug-in configuration create a `plugin-cfg.xml` file in ASCII format.

To use the Update global web server plug-in configuration page in the administrative console:

Procedure

1. Click **Environment > Update global web server plug-in configuration**.
2. Click **OK** to update the `plugin-cfg.xml` file.
3. Optional: Click **View or download the current web server plug-in configuration file** if you want to view or download the current version of this file.

   You can select this option if you want to:
   - View the current version of the file before you update it.
   - View the file after it is updated.
   - Download a copy of this file to a remote machine.

Results

Regenerating the configuration might take a while to complete. After it finishes, all objects in the administrative cell use their newest settings, which the web server can access. Whether triggered manually or occurring automatically, plug-in regeneration requires about 30 to 60 seconds to complete when the application server is on the same physical machine (node) as the web server. In other cases, it takes more time.

The delay is important because it determines how soon the new plug-in configuration takes effect. Suppose you add a new served path for a servlet, then regenerate the plug-in configurations. The regeneration requires 40 seconds, after which a user should be able to access the servlet by the new served path.

For an HTTP plug-in, the length of the delay is determined by the Refresh Interval attribute of the Config element in the `plugin-cfg.xml` file. The plug-in polls the disk, or file system, at this interval to see whether the configuration has changed. The default interval is 60 seconds. To regenerate the plug-in configuration requires twice the refresh interval.

In a development environment in which you are frequently changing settings in the administrative console, it is recommended that you set the refresh interval to 3 to 5 seconds.

In a production environment, set a longer refresh interval, perhaps as long as 30 minutes, depending on the frequency of changes.

What to do next

You might need to stop the application servers in the cell and then start the application servers again before the changes to the plug-in configuration go into effect.

If the web server is running on a remote machine, click **View or download the current web server plug-in configuration file** to download a copy of the `plugin-cfg.xml` file to a that machine.

When the deployment manager is installed on a machine that is remote from where the product is installed, one of the following solutions must be implemented in order for the `plugin-cfg.xml` file to retain the application server directory structures, and not assume those of the deployment manager after the plug-in is regenerated and a full synchronization occurs.

- **Command line**:

At a command prompt, enter the following command to change the `DeploymentManager/bin` directory and type on the machine where the deployment manager is installed. This command creates or updates the `plugin-cfg.xml` file, and changes all of the directories in the `plugin-cfg.xml` file to *app_server_root* directories.

```
GenPluginCfg -destination.root <app_server_root>
```

For example, issue the following command from the `DeploymentManager/bin` directory.

```
GenPluginCfg -destination.root "E:\WebSphere\AppServer"
```

- **plugin-cfg.xml file**:

  Edit the `plugin-cfg.xml` file to point to the correct directory structure for the log file, keyring, and stashfile.

  Perform a full synchronization so the `plugin-cfg.xml` file is replicated in all the nodes. You can use scripting or the administrative console to synchronize the nodes in the cell.

  The deployment manager `plugin-cfg.xml` file can point to the application server directories without any conflict.

Update the global web server plug-in configuration setting

Use this page to create or update a global plug-in configuration file. The configuration settings this file contains are based on the topology of the cell that contains the applications servers that use this web server plug-in. The web server plug-in configuration file settings determine whether an application server or the web server handles user requests.

A global web server plug-in configuration file must be regenerated whenever:

- You change the configuration settings for an application server, cluster, web container transport, or virtual host alias that is contained in the cell.
- You add a new application server, cluster, web container transport, or virtual host alias to the cell.

The generated `plugin-cfg.xml` file is placed in the `%was_profile_home%/config/cells` directory. If your web server is located on a remote machine, you must manually move this file to that machine.

Note: You can update the *global* `plugin-cfg.xml` file using the administrative console or running the GenPluginCfg command for all of the clusters in a cell. However, you must delete the `config/cells/plugin-cfg.xml` file before you update the *global* `plugin-cfg.xml` file. If you do not delete the `config/cells/plugin-cfg.xml` file, only the new properties and their values are added to the *global* `plugin-cfg.xml` file. Any updates to existing plug-in property values are not added to the *global* `plugin-cfg.xml` file.

To view this administrative console page, click **Environment > Update global web server plug-in configuration**.

Click **OK** to update the global `plugin-cfg.xml` file.

Click **View or download the current web server plug-in configuration file** if you want to:

- View the current version of the file before you update it.
- View the file after it is updated.
- Download a copy of this file to a remote machine.

Directory conventions

References in product information to *app_server_root*, *profile_root*, and other directories imply specific default directory locations. This topic describes the conventions in use for WebSphere Application Server.

Default product locations (distributed)

The following file paths are default locations. You can install the product and other components or create profiles in any directory where you have write access. Multiple installations of WebSphere Application Server Network Deployment products or components require multiple locations. Default values for installation actions by root and nonroot users are given. If no nonroot values are specified, then the default directory values are applicable to both root and nonroot users.

app_client_root

Table 14. Default installation root directories for the Application Client for IBM WebSphere Application Server.

This table shows the default installation root directories for the Application Client for IBM WebSphere Application Server.

User	Directory
Root	AIX `/usr/IBM/WebSphere/AppClient` (Java EE Application client only) HP-UX Linux Solaris `/opt/IBM/WebSphere/AppClient` (Java EE Application client only) Windows `C:\Program Files\IBM\WebSphere\AppClient`
Nonroot	AIX HP-UX Linux Solaris *user_home*`/IBM/WebSphere/AppClient` (Java EE Application client only) Windows `C:\IBM\WebSphere\AppClient`

app_server_root

Table 15. Default installation directories for WebSphere Application Server.

This table shows the default installation directories for WebSphere Application Server Network Deployment.

User	Directory
Root	AIX `/usr/IBM/WebSphere/AppServer` HP-UX Linux Solaris `/opt/IBM/WebSphere/AppServer` Windows `C:\Program Files\IBM\WebSphere\AppServer`
Nonroot	AIX HP-UX Linux Solaris *user_home*`/IBM/WebSphere/AppServer` Windows *user_home*`\IBM\WebSphere\AppServer`

component_root

The component installation root directory is any installation root directory described in this topic. Some programs are for use across multiple components—in particular, the Web Server Plug-ins, the Application Client, and the IBM HTTP Server. All of these components are part of the product package.

gskit_root

IBM Global Security Kit (GSKit) can now be installed by any user. GSKit is installed locally inside the installing product's directory structure and is no longer installed in a global location on the target system. The following list shows the default installation root directory for Version 8 of the GSKit, where *product_root* is the root directory of the product that is installing GSKit, for example IBM HTTP Server or the web server plug-in.

AIX HP-UX Linux Solaris

product_root`/gsk8`

Windows

product_root\gsk8

profile_root

Table 16. Default profile directories.

This table shows the default directories for a profile named profile_name on each distributed operating system.

User	Directory
Root	AIX /usr/IBM/WebSphere/AppServer/profiles/*profile_name* HP-UX Linux Solaris /opt/IBM/WebSphere/AppServer/profiles/*profile_name* Windows C:\Program Files\IBM\WebSphere\AppServer\profiles\profile_name
Nonroot	AIX HP-UX Linux Solaris *user_home*/IBM/WebSphere/AppServer/profiles Windows *user_home*\IBM\WebSphere\AppServer\profiles

plugins_root

Table 17. Default installation root directories for the Web Server Plug-ins.

This table shows the default installation root directories for the Web Server Plug-ins for WebSphere Application Server.

User	Directory
Root	AIX /usr/IBM/WebSphere/Plugins HP-UX Linux Solaris /opt/IBM/WebSphere/Plugins Windows C:\Program Files\IBM\WebSphere\Plugins
Nonroot	AIX HP-UX Linux Solaris *user_home*/IBM/WebSphere/Plugins Windows C:\IBM\WebSphere\Plugins

wct_root

Table 18. Default installation root directories for the WebSphere Customization Toolbox.

This table shows the default installation root directories for the WebSphere Customization Toolbox.

User	Directory
Root	AIX /usr/IBM/WebSphere/Toolbox HP-UX Linux Solaris /opt/IBM/WebSphere/Toolbox Windows C:\Program Files\IBM\WebSphere\Toolbox
Nonroot	AIX HP-UX Linux Solaris *user_home*/IBM/WebSphere/Toolbox Windows C:\IBM\WebSphere\Toolbox

web_server_root

Table 19. Default installation root directories for the IBM HTTP Server.

This table shows the default installation root directories for the IBM HTTP Server.

User	Directory
Root	AIX `/usr/IBM/HTTPServer` HP-UX Linux Solaris `/opt/IBM/HTTPServer` Windows `C:\Program Files\IBM\HTTPServer`
Nonroot	AIX HP-UX Linux Solaris *`user_home`*`/IBM/HTTPServer` Windows `C:\IBM\HTTPServer`

Configuring simple load balancing across multiple application server profiles with an administrative agent

Simple load balancing distributes HTTP requests across multiple IBM WebSphere Application Server instances. Also, you can configure simple load balancing to provide failover of an application state that is maintained in an HTTP session.

Before you begin

Note: This offering applies to stand-alone application server profiles for IBM WebSphere Application Server. This offering does not include a centralized management capability such as the deployment manager in WebSphere Application Server, Network Deployment.

About this task

You can configure simple load balancing capability with WebSphere Application Server by combining the plug-in configuration files of multiple stand-alone application server profiles into a single configuration file. The number of configuration files that you can combine are bound by the limits that exist in the WebSphere Application Server license agreement. You can use the following different configurations of the application server to combine the plug-in configuration files of multiple application server profiles into a single output file:

- Using multiple stand-alone base application server profiles. For more information, see the documentation on configuring simple load balancing across multiple application server profiles.
- Using multiple stand-alone base application server profiles with an administrative agent. This topic explains how to complete this configuration.
- Using multiple stand-alone base application server profiles with an administrative agent using the job manager. The job manager function is a part of WebSphere Application Server, Network Deployment. However, you can use the job manager function with stand-alone, base application server profiles. For more information, see the documentation on configuring simple load balancing across multiple stand-alone, base application server profiles with an administrative agent using the job manager.

Complete the following steps to register stand-alone application server profiles with an administrative agent and combine the plug-in configuration files from these profiles into a single output file.

Procedure

1. Install WebSphere Application Server and create application server profiles. For more information, see the documentation on WebSphere Application Server installation and application server profiles.
2. Configure the administrative agent and register each application server profile with the administrative agent. Complete the following steps:
   a. Set up the administrative agent, which includes creating the administrative agent profile.

b. Register the stand-alone application server with the administrative agent.

c. Start and stop the administrative agent.

After you complete these steps, you can complete all the administrative operations through the administrative agent. When you log in to the administrative console for the administrative agent, you can select which application servers to manage. For more information, see the documentation on administering stand-alone nodes using the administrative agent.

3. Install the enterprise application or web module. For more information, see the documentation on installing enterprise applications or modules.
4. Determine if you require session affinity.

   Session affinity directs requests from a given client to a specific application server. The application state maintained in the HTTP session is accessed in the HTTP session cache, which is local to the application server. Session affinity provides higher performance than database persistence of the session object, alone. Without session affinity, session requests must be obtained from the database if they are sent to a server that does not have the session object in the local cache.
5. Optional: Configure a unique HTTP session clone ID for *each* application server. You must complete this step if you require session affinity.

   You can configure a unique HTTP session clone ID using wsadmin scripting or the administrative console. To use wsadmin commands for the Jython or Jacl programming language, see the documentation on configuring a unique HTTP session clone ID for each application server using scripting. To configure a unique HTTP session clone ID using the administrative console, complete the following steps:

   a. Expand **Servers** > **Server Types** and click **WebSphere application servers** > *server_name*.

   b. Under **Container Settings**, expand **Web Container Settings**, and click **Web container**.

   c. Under **Additional Properties**, click **Custom properties** > **New**.

   d. In the **Name** field, enter `HttpSessionCloneId`.

   e. In the **Value** field, enter a unique value for the server. The unique value must be 8 - 9 alphanumeric characters; for example, test1234

   f. Click **Apply** or **OK**.

   g. Click **Save** to save the configuration changes to the master configuration.
6. Optional: Configure session persistence. If you require session failover capability, you must configure session persistence. Persistence of the session object to a database is the only option for session failover with WebSphere Application Server. To configure session persistence using the administrative console, see the documentation on configuring database session persistence. To configure database session persistence using wsadmin commands for the Jython or Jacl programming language, see the documentation on configuring database session persistence using scripting.
7. Restart the server.
8. Generate the `plugin-cfg.xml` file for each stand-alone application server using the GenPluginCfg script, the administrative console, or wsadmin scripting.

   To use the GenPluginCfg script, enter the following command on the command line:`*profile_root*/config/cells/GenPluginCfg.sh|bat`

   To use the administrative console, see the documentation on creating or updating a global web server plug-in configuration file.

   The following variables apply to the Jython and Jacl commands:

   - *cell_name* is the name of your cell.
   - *web_server_node* is the name of the node for your web server.
   - *web_server_name* is the name of your web server.

   Jython

   On the command line, enter each of the following commands on a separate line:

```
generator = AdminControl.completeObjectName('type=PluginCfgGenerator,*')
AdminControl.invoke(generator, 'generate', "profile_root/config cell_name web_server_node web_ser
```

Jacl On the command line, enter each of the following commands on a separate line:

```
set generator [$AdminControl completeObjectName type=PluginCfgGenerator,*]
$AdminControl invoke $generator generate "profile_root/config cell_name web_server_node web_serve
```

9. Merge the `plugin-cfg.xml` files from multiple application server nodes.

   You can either manually merge the `plugin-cfg.xml` files or use the pluginCfgMerge tool to automatically merge the `plugin-cfg.xml` file from multiple application server profiles into a single output file. The `pluginCfgMerge.bat` or `pluginCfgMerge.sh` tool is available after you install this fix pack and is located in the *`install_root`*`/bin` directory. To use the pluginCfgMerge tool, complete the following steps:

   a. Rename the `plugin-cfg.xml` files to a unique name across your application server profiles.
   b. Copy the `plugin-cfg.xml` file for all stand-alone application server profiles into a common directory.
   c. Use the pluginCfgMerge tool to combine the `plugin-cfg.xml` files from each of the application server profiles into a single output file. For example: AIX HP-UX Linux Solaris

      ```
      install_root/bin/pluginCfgMerge.sh plugin_configuration_file1 plugin_configuration_file2 resulting_plugin_config
      ```

      Windows

      ```
      install_root\bin\pluginCfgMerge.bat plugin_configuration_file1 plugin_configuration_file2 resulting_plugin_confi
      ```

      The *resulting_plugin_configuration_file* variable value is normally `plugin-cfg.xml`

   For more information about manually merging the `plugin-cfg.xml` files, see the technote on merging `plugin-cfg.xml` files from multiple application server profiles.

10. Copy the merged `plugin-cfg.xml` file to the *`plugin_installation_root`*`/config/`*`web_server_name`*`/` directory on the Web server host.
11. AIX HP-UX Linux Solaris Ensure that you have defined the correct operating system file access permissions for the merged `plugin-cfg.xml` file. These file access permissions allow the HTTP server plug-in process to read the file.

Results

When you complete this process, you have one plug-in configuration file for multiple stand-alone application server profiles.

Configuring simple load balancing across multiple application server profiles with an administrative agent using a job manager

Simple load balancing distributes HTTP requests across multiple IBM WebSphere Application Server instances. You can configure simple load balancing to provide failover of an application state that is maintained in an HTTP session.

About this task

You can configure simple load balancing capability with WebSphere Application Server by combining the plug-in configuration files of multiple stand-alone application server profiles into a single configuration file. The number of configuration files that you can combine are bound by the limits that exist in the WebSphere Application Server license agreement. You can use the following different configurations of the application server to combine the plug-in configuration files of multiple application server profiles into a single output file:

- Using multiple stand-alone base application server profiles. For more information, see the documentation on configuring simple load balancing across multiple application server profiles.

- Using multiple stand-alone base application server profiles with an administrative agent. For more information, see the documentation on configuring simple load balancing across multiple application server profiles with an administrative agent
- Using multiple stand-alone base application server profiles with an administrative agent using the job manager. The job manager function is a part of WebSphere Application Server, Network Deployment. However, you can use the job manager function with stand-alone, base application server profiles. Use this topic to complete this configuration.

Complete the following steps to register stand-alone application server profiles with an administrative agent using a job manager and combine the plug-in configuration files from these profiles into a single output file.

Procedure

1. Install WebSphere Application Server and create application server profiles. For more information, see the documentation on WebSphere Application Server installation and application server profiles.
2. Configure the administrative agent and register each application server profile with the administrative agent. Complete the following steps:
   a. Set up the administrative agent, which includes creating the administrative agent profile.
   b. Register the stand-alone application server with the administrative agent.
   c. Start and stop the administrative agent.

   After you complete these steps, you can complete all the administrative operations through the administrative agent. When you log in to the administrative console for the administrative agent, you can select which application servers to manage. For more information, see the documentation on administering stand-alone nodes using the administrative agent.
3. Install WebSphere Application Server, Network Deployment for the licensed WebSphere Application Server, Network Deployment instances that will perform centralized management for the stand-alone application server instances. For more information, see the installation documentation for the WebSphere Application Server, Network Deployment product.

   Attention: You must access the Information Center for WebSphere Application Server, Network Deployment to read its installation documentation.
4. Create the job manager profile, configure the job manager, and register stand-alone application servers with the job manager. For more information, see the documentation on setting up a job manager environment. You can complete administrative options centrally using the job manager. For more information about the job manager, see the conceptual information about the job manager.
5. Install the enterprise application or web module. You can use one of the following methods to install the enterprise application or web module:
   - Install the enterprise application or web module on each application server. For more information, see the documentation on installing enterprise applications or modules.
   - Install the enterprise application or web module using the job manager. For more information, see the documentation on installing applications using the job manager.
6. Determine if you require session affinity.

   Session affinity directs requests from a given client to a specific application server. The application state maintained in the HTTP session is accessed in the HTTP session cache, which is local to the application server. Session affinity provides higher performance than database persistence of the session object, alone. Without session affinity, session requests must be obtained from the database if they are sent to a server that does not have the session object in the local cache.
7. Optional: Configure a unique HTTP session clone ID for *each* application server. You must complete this step if you require session affinity.

   You can configure a unique HTTP session clone ID using wsadmin scripting or the administrative console. To use wsadmin commands for the Jython or Jacl programming language, see the documentation on configuring a unique HTTP session clone ID for each application server using scripting. To configure a unique HTTP session clone ID using the administrative console, complete the following steps:

a. Expand **Servers** > **Server Types** and click **WebSphere application servers** > *server_name*.
b. Under **Container Settings**, expand **Web Container Settings** and click **Web container**.
c. Under **Additional Properties**, click **Custom properties** > **New**.
d. In the **Name** field, enter `HttpSessionCloneId`.
e. In the **Value** field, enter a unique value for each server. The unique value must be 8 - 9 alphanumeric characters; for example, test1234
f. Click **Apply** or **OK**.
g. Click **Save** to save the configuration changes to the master configuration.

8. Optional: Configure session persistence. If you require session failover capability, you must configure session persistence. Persistence of the session object to a database is the only option for session failover with WebSphere Application Server. To configure session persistence using the administrative console, see the documentation on configuring database session persistence. To configure database session persistence using wsadmin commands for the Jython or Jacl programming language, see the documentation on configuring database session persistence using scripting.
9. Restart the server.
10. Generate the `plugin-cfg.xml` file for each stand-alone application server using the GenPluginCfg script, the administrative console, or wsadmin scripting.

    To use the GenPluginCfg script, enter the following command on the command line:`*profile_root*/config/cells/GenPluginCfg.sh|bat`

    To use the administrative console, see the documentation on creating or updating a global web server plug-in configuration file.

    The following variables apply to the Jython and Jacl commands:
    - *cell_name* is the name of your cell.
    - *web_server_node* is the name of the node for your web server.
    - *web_server_name* is the name of your web server.

    Jython
    On the command line, enter each of the following commands on a separate line:

    ```
    generator = AdminControl.completeObjectName('type=PluginCfgGenerator,*')
    AdminControl.invoke(generator, 'generate', "profile_root/config cell_name web_server_node web_ser
    ```

    Jacl On the command line, enter each of the following commands on a separate line:

    ```
    set generator [$AdminControl completeObjectName type=PluginCfgGenerator,*]
    $AdminControl invoke $generator generate "profile_root/config cell_name web_server_node web_serve
    ```

11. Merge the `plugin-cfg.xml` files from multiple application server nodes.

    You can either manually merge the `plugin-cfg.xml` files or use the tool to automatically merge the `plugin-cfg.xml` file from multiple application server profiles into a single output file. The `.bat` or `.sh` tool is available after you install this fix pack and is located in the *install_root*`/bin` directory. To use the tool, complete the following steps:

    a. Rename the `plugin-cfg.xml` files to a unique name across your application server profiles.
    b. Copy the `plugin-cfg.xml` file for all stand-alone application server profiles into a common directory.
    c. Use the tool to combine the `plugin-cfg.xml` files from each of the application server profiles into a single output file. For example: AIX HP-UX Linux Solaris

    ```
    install_root/bin/.sh plugin_configuration_file1 plugin_configuration_file2 resulting_plugin_configuration_file
    ```

    Windows

    ```
    install_root\bin\.bat plugin_configuration_file1 plugin_configuration_file2 resulting_plugin_configuration_file
    ```

    The *resulting_plugin_configuration_file* variable value is normally `plugin-cfg.xml`

For more information about manually merging the `plugin-cfg.xml` files, see the technote on merging `plugin-cfg.xml` files from multiple application server profiles.

12. Copy the merged `plugin-cfg.xml` file to the *`plugin_installation_root`*`/config/`*`web_server_name`*`/` directory on the Web server host.
13. AIX HP-UX Linux Solaris Ensure that you have defined the correct operating system file access permissions for the merged `plugin-cfg.xml` file. These file access permissions allow the HTTP server plug-in process to read the file.

Results

When you complete this process, you have one plug-in configuration file for multiple stand-alone application server profiles.

Administering web server plug-ins

Using the same HTTP server to handle HTTP and HTTPS requests for multiple cells

In a multiple cell environment you might want to use the same HTTP server to send and receive HTTP and HTTPS requests for multiple cells. To accomplish this setup, you must merge the web server plug-in configuration files that are being used by the application servers in those cells into a single web server plug-in configuration file.

About this task

You can use the following different configurations of the application server to combine the plug-in configuration files of multiple WebSphere Application Server, Network Deployment cells into a single configuration file.

This technique can also be employed to merge all of the web server plug-in configuration files into a single configuration file when you are migrating to a new release of the product and need to route traffic to multiple cells running an older version of WebSphere Application Server, Network Deployment as well as the new version.

To create a single plug-in configuration file for all of your cells, you must first create a separate plug-in configuration file for each of the cells, and then combine these files into a single configuration file. The web server plug-in for each cell then uses the same web server plug-in configuration file.

Complete the following steps to merge multiple web server plug-in configuration files into a single configuration file.

Procedure

1. Use either the GenPluginCfg script, the administrative console, or wsadmin scripting to generate the `plugin-cfg.xml` file for each cell.

   See the topic *GenPluginCfg command* for a description of how to use the GenPluginCfg script to generate the `plugin-cfg.xml` file.

   See the topic *Implementing a web server plug-in* for a description of how to use the administrative console to generate the `plugin-cfg.xml` file.

   See the topic *Regenerating the node plug-in configuration using scripting* for a description of how to use wsadmin scripting to generate the `plugin-cfg.xml` file.
2. Merge the `plugin-cfg.xml` files from multiple application server cells.

You can either manually merge the `plugin-cfg.xml` files or use the pluginCfgMerge tool to automatically merge the `plugin-cfg.xml` file from multiple application server profiles into a single output file. The `pluginCfgMerge.bat` and `pluginCfgMerge.sh` files are located in the *install_root*`/bin` directory.

To use the pluginCfgMerge tool, complete the following steps:

a. Rename the `plugin-cfg.xml` files to a unique name across your application server cells.

b. Copy the `plugin-cfg.xml` file for all application server cell profiles into a common directory.

c. Use the pluginCfgMerge tool to combine the `plugin-cfg.xml` files from each of the application server cells into a single output file. For example:

   AIX HP-UX Linux Solaris

   ```
   install_root/bin/pluginCfgMerge.sh plugin_configuration_file1 plugin_configuration_file2 resulting_plugin_configu
   ```

   Windows

   ```
   install_root\bin\pluginCfgMerge.bat plugin_configuration_file1 plugin_configuration_file2 resulting_plugin_config
   ```

   The *resulting_plugin_configuration_file* variable value is normally `plugin-cfg.xml`

For more information about manually merging the `plugin-cfg.xml` files, see the technote on merging `plugin-cfg.xml` files from multiple application server profiles.

3. Ensure that the cloneID value for each application server is unique.

   Examine the cloneID value for each application server in the merged file to ensure that this value is unique for each application server. If the cloneID values in the merged file are not all unique, or if you are running with memory to memory session replication in peer to peer mode, use wsadmin scripting or the administrative console to configure unique HTTP session cloneIDs.

   To use wsadmin commands for the Jython or Jacl programming language, see the documentation on configuring a unique HTTP session clone ID for each application server using scripting.

   To configure a unique HTTP session clone ID using the administrative console, complete the following steps:

   a. Click **Servers > Server Types > WebSphere application servers >** *server_name*.

   b. Under Container Settings, click **Web Container Settings > Web container**.

   c. Under Additional Properties, click **Custom properties > New**.

   d. Enter `HttpSessionCloneId` in the **Name** field, and a unique value for the server in the **Value** field.

      The unique value must be 8 - 9 alphanumeric characters in length. For example, test1234 is a valid cloneID value.

   e. Click **Apply** or **OK**.

   f. Click **Save** to save the configuration changes to the master configuration.

4. Copy the merged `plugin-cfg.xml` file to the *plugin_installation_root*`/config/`*web_server_name*`/` directory on the web server host.

5. AIX HP-UX Linux Solaris Ensure that you have defined the correct operating system file access permissions for the merged `plugin-cfg.xml` file. These file access permissions allow the HTTP server plug-in process to read the file.

Results

When you complete this process, you have one plug-in configuration file for multiple application server cells, and can use the same HTTP server to handle HTTP and HTTPS requests for multiple cells.

Web server plug-in properties

Use this page to view or change the settings of a web server plug-in configuration file. The plug-in configuration file, `plugin_cfg.xml`, provides properties for establishing communication between the Web server and the Application Server.

To view this administrative console page, click **Servers > Server Types > Web servers >** *web_server_name* **> Plug-in Properties**.

On the **Configuration** tab, you can edit fields. On the **Runtime** tab, you can look at read-only information.

The **Runtime** tab is available only when this web server has accessed applications running on application servers and there is an `http_plugin.log` file.

Note: This topic references one or more of the application server log files. Beginning in WebSphere Application Server Version 8.0 you can configure the server to use the High Performance Extensible Logging (HPEL) log and trace infrastructure instead of using `SystemOut.log` , `SystemErr.log`, `trace.log`, and `activity.log` files or native z/OS logging facilities. If you are using HPEL, you can access all of your log and trace information using the LogViewer command-line tool from your server profile bin directory. See the information about using HPEL to troubleshoot applications for more information on using HPEL.

Ignore DNS failures during web server startup

Specifies whether the plug-in ignores DNS failures within a configuration when starting.

This field corresponds to the IgnoreDNSFailures element in the plugin-cfg.xml file.

When you set the value to **true**, the plug-in ignores DNS failures within a configuration and starts successfully if at least one server in each ServerCluster is able to resolve the host name. If a server host name cannot be resolved, it is marked **unavailable** for the continued existence of the configuration. Further attempts to resolve the host name are not made during the routing of requests. If a DNS failure occurs, a message is written to the plug-in log file and the plug-in initialization process continues.

By default, when the value is **false**, DNS failures cause the plug-in to not initialize and requests fail. However, the web server starts.

Data type	String
Default	false

Refresh configuration interval

Specifies the time interval, in seconds, at which the plug-in should check the configuration file to see if updates or changes have occurred. The plug-in checks the file for any modifications that have occurred since the last time the plug-in configuration was loaded.

In a development environment in which changes are frequent, a lower setting than the default setting of 60 seconds is preferable. In production, a higher value than the default is preferable because updates to the configuration will not occur so often. If the plug-in reload fails for some reason, a message is written to the plug-in log file and the previous configuration is used until the plug-in configuration file successfully reloads. If you are not seeing the changes you made to your plug-in configuration, check the plug-in log file for indications of the problem.

Data type	Integer
Default	60 seconds.

Plug-in configuration file name

Specifies the file name of the configuration file for the plug-in. The Application Server generates the `plugin-cfg.xml` file by default. The configuration file identifies applications, Application Servers, clusters, and HTTP ports for the web server. The web server uses the file to access deployed applications on various Application Servers.

Note: This field is disabled and cannot be changed, but the value is displayed for information purposes only.

If a different location is desired, you need to rerun the Plugin installer to define the new location, and then run the new **configureWebserver** script that is produced from the install process on your WebSphere Application Server machine.

If you select a web server plug-in during installation, the installer program configures the web server to identify the location of the `plugin-cfg.xml` file, if possible. The plug-in configuration file, by default, is installed in the *plugins_root*/`config`/*`web_server_name`* directory.

The installer program adds a directive to the web server configuration that specifies the location of the `plugin-cfg.xml` file.

For remote web servers, you must copy the file from the local directory where the Application Server is installed to the remote machine. This is known as propagating the plug-in configuration file. If you are using IBM HTTP Server V6.1 or higher for your web server, WebSphere Application Server can automatically propagate the plug-in configuration file for you to remote machines provided there is a working HTTP transport mechanism to propagate the file.

You can click **View** to display a copy of the current plug-in configuration file.

Data type	String
Default	`plugin-cfg.xml`

Automatically generate plug-in configuration file

To automatically generate a plug-in configuration file to a remote web server:

- This field must be checked.
- The plug-in configuration service must be enabled

When the plug-in configuration service is enabled, a plug-in configuration file is automatically generated for a web server whenever:

- The WebSphere Application Server administrator defines new web server.
- An application is deployed to an Application Server.
- An application is uninstalled.
- A virtual host definition is updated and saved.

By default, this field is checked. Clear the check box if you want to manually generate a plug-in configuration file for this web server.

Important: When the plug-in configuration file is generated, it does not include admin_host on the list of virtual hosts. The Information Center article "Allowing web servers to access the administrative console" describes how to add admin_host to the list of virtual hosts.

Automatically propagate plug-in configuration file

Specifies whether or not you want the application server to automatically propagate a copy of a changed plug-in configuration file to a Web server:

- This field must be checked.
- The plug-in configuration service must be enabled
- In a Network Deployment environment, a WebSphere Application Server node agent must be on the node that hosts the web server associated with the changed plug-in configuration file.

By default, this field is checked.

Note: The plug-in configuration file can only be automatically propagated to a remote web server if that web server is an IBM HTTP Server V6.1 or higher web server and its administration server is running.

Because the plug-in configuration service runs in the background and is not tied to the administrative console, the administrative console cannot show the results of the automatic propagation.

For distributed platforms, you can check the related messages in the deployment manager SystemOut.log file to verify that the automatic propagation successfully completed.

Plug-in key store file name

Specifies the fully qualified directory path and file name of the database file containing your security key rings that the Web server plug-in uses for HTTPS requests. This file resides on the Web server that is associated with this web server plug-in. After you specify the fully qualified directory path and file name of the database file, you can:

- Cick **Manage keys and certificates** to update this file.
- Click **Copy to web server key store directory** to add a copy of this file to the key store directory for the web server.

Data type	String
Default	None

Plug-in configuration directory and file name

Specifies the fully qualified path of the web server copy of the web server plug-in configuration file. This path is the name of the file and its location on the machine where the web server is running.

Note: This field is disabled and cannot be changed, but the value is displayed for information purposes only.

Plug-in key store directory and file name

Specifies the fully qualified path of the web server copy of the database file that contains your security key rings. This path is the name of the file and its location on the machine where the web server is running.

Note: This field is disabled and cannot be changed, but the value is displayed for information purposes only.

Plug-in logging

Specifies the location and name of the `http_plugin.log` file. Also specifies the scope of messages in the log.

The log describes the location and level of log messages that are written by the plug-in. If a log is not specified within the configuration file, then, in some cases, log messages are written to the web server error log.

On a distributed platform, if the log file does not exist then it will be created. If the log file already exists, it will be opened in append mode and the previous plug-in log messages will remain.

Log file name - The fully qualified path to the log file to which the plug-in will write error messages.

Data type	String
Default	*plugins_root*/`logs`/*web_server_name*/`http_plugin.log` Specify the file path of the `http_plugin.log` file.

Log level- The level of detail of the log messages that the plug-in should write to the log. You can specify one of the following values for this attribute:
- Trace. All of the steps in the request process are logged in detail.
- Stats. The server selected for each request and other load balancing information relating to request handling is logged.
- Warn. All warning and error messages resulting from abnormal request processing are logged.
- Error. Only error messages resulting from abnormal request processing are logged.
- Debug. All of the critical steps performed in processing requests are logged.
- Detail. All of the information about requests and responses are logged.

If a Log level is not specified, the default value **Error** is used.

Be careful when setting the level to **Trace**. A lot of messages are logged at this level which can cause the disk space/file system to fill up very quickly. A **Trace** setting should never be used in a normally functioning environment as it adversely affects performance.

Important: If the web server and web server plug-in are running on an AIX, HP-UX, Linux, or Solaris system, and you change the log level, in the plugin-cfg.xml file, this change is not picked up dynamically. You must restart the web server to pick up the change. For example on Solaris, if you do not restart the web server, the following error message appears in the *Plugin_Home*/logs/http_plugin.log file:

```
ERROR: ws_config_parser:handleLogEnd: Failed to open log file
'/opt/IBM/WebSphere/Plugin/logs/sunwebserver/http_plugin.log', OS
```

Data type	String
Default	Error

Web server plug-in request and response optimization properties

Use this page to view or change the request and response optimization properties for a web server plug-in.

To view this administrative console page, click **Servers** > **Server Types** > **Web servers** > *web_server_name* > **Plug-in properties** > **Request and response**.

Maximum chunk size used when reading the HTTP response body:

Specifies the maximum chunk size the plug-in can use when reading the response body.

This field corresponds to the ResponseChunkSize element in the plugin-cfg.xml file.

The plug-in reads the response body in 64K chunks until all of the response data is read. This approach causes a performance problem for requests whose response body contains large amounts of data.

If the content length of the response body is unknown, the values specified for this property is used as the size of the buffer that is allocated. The response body is then read in this size chunks, until the entire body is read. If the content length is known, then a buffer size of either the content length or the specified size (whichever is less) is used to read the response body.

Data type	Integer
Default	64 kilobytes Specify the size in kilobytes (1024 byte blocks).

Enable Nagle algorithm for connections to the Application Server:

When checked, the Nagle algorithm is enabled for connections between the plug-in and the Application Server.

This field corresponds to the ASDisableNagle element in the plugin-cfg.xml file.

The Nagle algorithm is named after engineer John Nagle, who invented this standard part of the transmission control protocol/internet protocol (TCP/IP). The algorithm reduces network overhead by adding a transmission delay (usually 200 milliseconds) to a small packet, which lets other small packets arrive and be included in the transmission. Because communications has an associated cost that is not as dependent on packet size as it is on frequency of transmission, this algorithm potentially reduces overhead with a more efficient number of transmissions.

By default, this field is not checked, and the Nagle algorithm is disabled. Select this field to enable the Nagle algorithm.

Enable Nagle Algorithm for the IIS Web Server:

When checked, the Nagle algorithm is used for connections from the Microsoft Internet Informations Services (IIS) Web Server to the application server.

This field corresponds to the IISDisableNagle element in the plugin-cfg.xml file. It only applies if you are using the Microsoft Internet Informations Services (IIS) Web Server.

By default, this field is not checked, and the Nagle algorithm is disabled. Select this field to enable the Nagle algorithm for this connection.

Chunk HTTP response to the client:

When checked, responses to the client are broken into chunks if a `Transfer-Encoding : Chunked` response header is present in the response.

This field corresponds to the ChunkedResponse element in the plugin-cfg.xml file. It only applies if you are using a Microsoft Internet Informations Services (IIS) Web Server, a Java System web server, or a Domino Web Server. The IBM HTTP Server automatically handles breaking the response into chunks to send to the client.

By default, this field is not checked, and responses are not broken into chunks. Select this field to enable responses to the client to be broken into chunks if a `Transfer-Encoding : Chunked` response header is present in the response.

Accept content for all requests: This field corresponds to the AcceptAllContent element in the plugin-cfg.xml file.

When selected, you can include content in GET, and HEAD requests when a Content-Length or Transfer-encoding header is contained in the request header.

By default, this field is checked.

Select this field to enable users to include content in GET, and HEAD requests when a Content-Length or Transfer-encoding header is contained in the request header.

Note: The **Accept content for all requests** setting on the administrative console corresponds to the AcceptAllContent attribute in the plugin-cfg.xml file. For Version 8, the default for the setting is checked and for the attribute is true. Before Version 8, the default for the setting is not checked and for the attribute is false.

Virtual host matching:

When selected, virtual host mapping is performed by physically using the port number for which the request was received.

This field corresponds to the VHostMatchingCompat element in the plugin-cfg.xml file.

By default, this field is not checked, and matching is done logically using the port number contained in the host header. Select this field if you want virtual host mapping performed by physically using the port number for which the request was received.

Use the radio buttons to make your physical or logical port selection.

Application server port preference:

Specifies which port number the Application Server should use to build URIs for a sendRedirect. This field is only applicable for a sendRedirect if you use relative URIs and does not affect absolute redirects. This field also specifies where to retrieve the value for HttpServletRequest.getServerPort().

This field corresponds to the AppServerPortPreference element in the plugin-cfg.xml file.

Specify:
- `hostHeader` if the port number from the host header of the HTTP request coming in is to be used.
- `webserverPort` if the port number on which the web server received the request is to be used.

The default is `hostHeader`.

Web server plug-in caching properties

Use this page to view or change the caching properties for a web server plug-in.

To view this administrative console page, click **Servers** > **Server Types** > **Web servers** > *web_server_name* > **Plug-in properties** > **Caching**.

Enable Edge Side Include (ESI) processing to cache the responses:

Specifies whether to enable Edge Side Include processing to cache the responses.

This field corresponds to the esiEnable element in the plugin-cfg.xml file.

By default, this field is checked. Select this field if you want Edge Side Include (ESI) processing used to cache responses. If ESI processing is disabled for the plug-in, the other ESI plug-in properties are ignored. Clear the checkbox to disable Edge Side Include processing.

Enable invalidation monitor to receive notifications:

When checked, the ESI processor receives invalidations from the application server.

This field corresponds to the ESIInvalidationMonitor element in the plugin-cfg.xml file. It is ignored if Edge Side Include (ESI) processing is not enabled for the plug-in.

By default, this field is not selected. Clear the check box if you do not want the application server to send invalidations to the ESI processor.

Maximum cache size:

Specifies, in 1K byte units, the maximum size of the cache. The default maximum size of the cache is 1024K bytes (1 megabyte). If the cache is full, the first entry to be evicted from the cache is the entry that is closest its expiration time.

This field corresponds to the esiMaxCacheSize element in the plugin-cfg.xml file.

Data type	Integer
Default	1024 kilobytes Specify the size in kilobytes (1024 byte blocks).

Web server plug-in request routing properties

Use this page to view or change the request routing properties for a web server plug-in.

To view this administrative console page, click **Servers** > **Server Types** > **Web servers** > *web_server_name* > **Plug-in properties** > **Request routing**.

Load balancing option:

Specifies the load balancing option that the plug-in uses in sending requests to the various application servers associated with that web server.

This field corresponds to the LoadBalance element in the plugin-cfg.xml file.

Select the appropriate load balancing option:

- Round robin
- Random

The Round Robin implementation has a random starting point. The first application server is picked randomly. Round Robin is then used to pick application servers from that point forward. This implementation ensures that in multiple process based web servers, all of the processes don't start up by sending the first request to the same Application Server.

The default load balancing type is Round Robin.

Retry interval:

Specifies the length of time, in seconds, that should elapse from the time an application server is marked down to the time that the plug-in retries a connection.

This field corresponds to the RetryInterval element in the plugin-cfg.xml file.

Data type	Integer
Default	60 seconds

Maximum size of request content:

Select whether there is a limit on the size of request content. If limited, this field also specifies the maximum number of kilobytes of data a request can contain. When a limit is set, the plug-in fails any request that is received that contains more data than the specified limit.

This field corresponds to the PostSizeLimit element in the plugin-cfg.xml file.

Select whether to limit the size of request content:

- `No limit`

- `Set limit`

If you select `Set limit`, specify a limit size.

Data type	Integer Specify the size in kilobytes (1024 byte blocks).
Default	`-1`, which indicates there is no limit for the post size.

Maximum buffer size used when reading HTTP request content:

Specifies, in kilobytes, the maximum buffer size that is used when the content of an HTTP request is read. If the application server that initially receives a request cannot process that request, the data contained in this buffer is sent to another application server in an attempt to have that application server process the request.

This field corresponds to the PostBufferSize element in the plugin-cfg.xml file.

If **Set limit** is selected, specify a limit size.

Data type	Integer Specify the size in kilobytes (1024 byte blocks).
Default	`64`

Remove special headers:

When checked, the plug-in will remove any headers from incoming requests before adding the headers the plug-in is supposed to add before forwarding the request to an application server.

This field corresponds to the RemoveSpecialHeaders element in the plugin-cfg.xml file.

The plug-in adds special headers to the request before it is forwarded to the application server. These headers store information about the request that will need to be used by the application. Not removing the headers from incoming requests introduces a potential security exposure.

By default, the special headers are not retained. Clear the check box to retain special headers.

Clone separator change:

When this option is selected, the plug-in expects the plus character (+) as the clone separator.

This field corresponds to the ServerCloneID element in the plugin-cfg.xml file.

Some pervasive devices cannot handle the colon character (:) used to separate clone IDs in conjunction with session affinity. If this field is checked, you must also change the configurations of the associated application servers such that the application servers separates clone IDs with the plus character as well.

By default, this option is selected. Clear the field if you want to use the colon character to separate clone IDs.

Web server plug-in configuration service property

Use this page to view or change the configuration settings for the web server plug-in configuration service.

Note: This topic references one or more of the application server log files. Beginning in WebSphere Application Server Version 8.0 you can configure the server to use the High Performance Extensible Logging (HPEL) log and trace infrastructure instead of using `SystemOut.log` , `SystemErr.log`, `trace.log`, and `activity.log` files or native z/OS logging facilities. If you are using HPEL, you can access all of your log and trace information using the LogViewer command-line tool from your server profile bin directory. See the information about using HPEL to troubleshoot applications for more information on using HPEL.

If you are using a stand-alone application server, click **Servers > Server Types > WebSphere application servers >** *server_name* **> Administration Services > Web server plug-in configuration service** to view this administrative console page.

For a WebSphere Application Server, Network Deployment configuration, if you are using the deployment manager, click **System Administration > Deployment manager > Administration Services > Web server plug-in configuration service**.

For IBM i and distributed platforms running in a WebSphere Application Server, Network Deployment environment, the deployment manager SystemOut log contains the status of the automatic plug-in generation and propagation.

For IBM i and distributed platforms running in a Base or Express® environment, the application server's SystemOut log contains the status of the automatic plug-in generation and propagation.

Enable automated web server configuration processing

The web server plug-in configuration service is selected by default. The service automatically generates the plug-in configuration file whenever the web server environment changes, with a few exceptions. For example, the plug-in configuration file is regenerated whenever one of the following activities occurs:

- A new application is deployed on an associated application server
- The web server definition is saved
- An application is removed from an associated application server
- A new virtual host is defined

In WebSphere Application Server, Network Deployment configurations, the plug-in configuration file does not regenerate when:

- A cluster member is added to a cluster.
- TCP channel settings are updated for an application server.

By default, this option is selected. Clear the field to disable automated web server configuration processing.

Application Server property settings for a web server plug-in

Use this page to view or change application server settings for a web server plug-in.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers >** *server_name*, and then, in the Additional Properties section, click **Web server plug-in properties**.

Server Role

Specifies the role this application server is assigned.

Select `Primary` to add this application server to the list of primary application servers. The plug-in initially attempts to route requests to the application servers on this list.

Select `Backup` to add this application server to the list of backup application servers. The plug-in does not load balance across the backup application servers. A backup server is only used if a primary server is not available. When the plug-in determines that a backup application server is required, it goes through the list of backup servers, in order, until no servers are left in the list or until a request is successfully sent and a response received from one of the servers on this list.

Default setting	Primary

Connection timeout

Specifies whether there is a limited amount of time during which the application server maintains a connection with the web server.

This field corresponds to the ConnectTimeout element in the plugin-cfg.xml file.

The setting for this field determines whether the plug-in performs non-blocking connections with the application server. Non-blocking connections are beneficial when the plug-in is unable to contact the destination to determine whether or not the port is available.

If the **Use connection timeout** setting is not selected, the plug-in performs nonblocking connections with the application server. If the **Use connection timeout** setting is selected, you must specify a value in the **seconds** field:

- If you specify a value greater than 0 in the **seconds** field, the plug-in waits the specified number for seconds to perform a successful connection. If a connection does not occur during that time interval, the plug-in marks the server unavailable, and sends the request to another application server in the cluster.
- If you specify a value of 0 in the **seconds** field, the plug-in performs a blocking connection.
- If you do not specify a value in the **seconds** field, the plug-in performs a blocking connection where the plug-in sits until the operating system times out, whhich might be as long as two minutes, depending on the platform, before it marks the server `unavailable`.

Data type	Integer
Default	5

Use read/write timeout

Specifies whether there is a time limit for how long the plug-in waits to send a request to or receive a response from the application server.

This field corresponds to the ServerIOTimeout element in the plugin-cfg.xml file.

Select the **Use read/write timeout** property to set a read/write timeout. When you select this setting, you must specify the length of time, in seconds that the plug-in waits to send a request or to receive a response. When selecting a value to specify for this field, remember that it might take a couple of minutes for an application server to process a request. Setting the value too low might cause the plug-in to send a false server error response to the client. If the checkbox is not checked, the plug-in uses blocked I/O to write requests to and read responses from the application server until the TCP connection times out.

Note: The **Use read/write timeout** setting on the administrative console corresponds to the ServerIOTimeout attribute in the plugin-cfg.xml file. The default value for this setting is different from the default value in previous versions of the product.

Data type	Integer
Default	900 seconds

Use maximum number of connections

Specifies the maximum number of pending connections to an Application Server that can be flowing through a web server process at any point in time.

This field corresponds to the ServerMaxConnections element in the plugin-cfg.xml file.

Select the **Use maximum number of connections** property to set a maximum number of connections. When you select this setting, you, must specify the maximum number of connections that can exist between the web server and the Application Server at any given point in time.

For example, assuming that:

- The application server is fronted by 5 nodes that are running an IHS web server.
- Each node starts 2 processes.
- This property is set to 50.

In this example, the application server could potentially get up to 500 connections. (You take the number of nodes, 5, multiply it by the number of processes, 2, and then multiply that number by the number specified for this property, 50, for a total of 500 connections.

If this property is set to either zero or -1, there is no limit to the number of pending connections to the Application Servers.

Data type	Integer
Default	0

Use extended handshake to check whether application server is running

When selected, the web server plug-in uses an extended handshake to check whether the application server is running.

This field corresponds to the ExtendedHandshake element in the plugin-cfg.xml file.

Select this property if a proxy firewall is between the plug-in and the application server.

The plug-in marks a server as down when the connect() fails. However, when a proxy firewall is in between the plug-in and the application server, the connect() will succeed, even though the back end application server is down. This causes the plug-in to not failover correctly to another application server.

If the plug-in performs some handshaking with the application server to ensure that the application server is started before the plug-in sends a request, the plug-in can failover to another application server if it detects that the application server with which it is attempting to perform a handshake is not available.

By default, this field is not selected. Select this field if you want to use extended handshake to check whether an application server is running.

Send the header "100 Continue" before sending the request content

Specifies whether the web server plug-in sends the header "100 Continue" to the application server before it sends the request content.

This field corresponds to the WaitForContinue element in the plugin-cfg.xml file.

When selected, the web server plug-in sends the header "100 Continue" to the application server before it sends the request content.

By default, this field is not selected. Select this field to enable this function.

plugin-cfg.xml file

There are two types of `plugin-cfg.xml` files: application-centric and topology-centric.

An application-centric file has an application that is mapped to both web server and application server definitions. Changes that you make to the plug-in by using the administrative console persist to the `plugin-cfg.xml` file upon generation.

A topology-centric file represents everything that is defined in the environment. Typically, this `plugin-cfg.xml` file is used when you do not have web servers defined. Consider the following rules when you want to update a topology-centric `plugin-cfg.xml` file:

- If the `plugin-cfg.xml` file *exists* within the *`app_server_root`*`\dmgr\cells` directory, the plug-in generation process ignores the updated values from the **application server** > **Web Server Plugin Properties** panel of the administrative console and uses the existing values within the XML file. In this case, you must manually update the XML file for those values to persist.
- If the `plugin-cfg.xml` file *does not exist* within the *`app_server_root`*`\dmgr\cells` directory, the plug-in generation process creates a new `plugin-cfg.xml` file. The process persists the latest values that are set on the **Application Server** > **Web Server Plugin Properties** panel within the administrative console.

The design of this file and its related function enable the product to support different types of configurations. For example, web server definitions might not exist. In addition, many web server plug-in properties, such as RefreshInterval, LogLevel, and the Edge Side Include (ESI) processor properties, can be updated manually only. Those values must be maintained through each iteration.

The `plugin-cfg.xml` file includes the following elements and attributes. Unless indicated otherwise, you can specify each element and attribute only once within the `plugin-cfg.xml` file.

gotcha: Use the administrative console to set these properties for each web server definition. Any manual changes you make to the plug-in configuration file for each web server are overridden whenever the file is regenerated.

Note: You can update the *global* `plugin-cfg.xml` file using the administrative console or running the GenPluginCfg command for all of the clusters in a cell. However, you must delete the `config/cells/plugin-cfg.xml` file before you update the *global* `plugin-cfg.xml` file. If you do not delete the `config/cells/plugin-cfg.xml` file, only the new properties and their values are added to the *global* `plugin-cfg.xml` file. Any updates to existing plug-in property values are not added to the *global* `plugin-cfg.xml` file.

The file can include one or more of the following elements and attributes. The `Config` element is required.

- “Config” on page 162
- “IgnoreDNSFailures” on page 162
- “RefreshInterval” on page 162
- “ASDisableNagle” on page 162
- Windows “IISDisableNagle” on page 163
- “VHostMatchingCompat” on page 163
- “AppServerPortPreference” on page 163
- “ResponseChunkSize” on page 163
- “AcceptAllContent” on page 163
- “ChunkedResponse” on page 163
- Windows “IISPluginPriority” on page 164
- “TrustedProxyEnable” on page 164
- “TrustedProxyList” on page 164

Config

This element, which is required, starts the HTTP plug-in configuration file.

IgnoreDNSFailures

Specifies whether the plug-in ignores DNS failures within a configuration when starting. When set to `true`, the plug-in ignores DNS failures within a configuration and starts successfully if at least one server in each server cluster is able to resolve the host name. Any server for which the host name cannot be resolved is marked *unavailable* for the life of the configuration. No attempts to resolve the host name are made during the routing of requests. If a DNS failure occurs, a log message is written to the plug-in log file, and the plug-in initialization continues rather than causing the web server not to start. The default value is `false`, meaning DNS failures cause the web server not to start.

RefreshInterval

Specifies the time interval, in seconds, at which the plug-in checks the configuration file to see if updates or changes have occurred. The plug-in checks the file for any modifications that have occurred since the last time the plug-in configuration was loaded.

In a development environment in which changes are frequent, a lower setting than the default setting of 60 is preferable. In production, a higher value than the default is preferable because updates to the configuration do not occur so often. If the plug-in reload fails for some reason, a message is written to the plug-in log file and the previous configuration is used until the plug-in configuration file successfully reloads. If you are not seeing the changes you made to your plug-in configuration, check the plug-in log file for indications of the problem.

ASDisableNagle

Specifies whether the user wants to disable the nagle algorithm for the connection between the plug-in and the application server. By default, the nagle algorithm is enabled.

The value can be `true` or `false`.

Windows

IISDisableNagle

Specifies whether the user wants to disable the nagle algorithm on Microsoft Internet Information Services (IIS). By default, the nagle algorithm is enabled.

The value can be `true` or `false`.

VHostMatchingCompat

Specifies that the port number is to be used for virtual host matching. Specify one of the following values:
- `true` if matching is to be done physically by using the port number for which the request was received.
- `false` if matching is to be done logically by using the port number contained in the host header.

The default value is `false`.

AppServerPortPreference

Specifies which port number the application server uses to build URIs for a sendRedirect() method. The following values can be specified:
- `hostHeader` if the port number from the host header of the HTTP request coming in is to be used.
- `webserverPort` if the port number on which the web server received the request is to be used.

The default value is `hostHeader`.

ResponseChunkSize

Specifies the maximum chunk size to use when reading the response body. For example, specify `Config ResponseChunkSize="N">`, where *N* equals the chunk size in kilobytes.

The plug-in reads the response body in 64 K chunks until all of the response data is read. This approach causes a performance problem for requests whose response body contains large amounts of data.

If the content length of the response body is unknown, a buffer size of *N* KBs is allocated and the body is read in *N* KB size chunks, until the entire body is read. If the content length is known, then a buffer size of either content length or *N* (whichever is less) is used to read the response body.

The default chunk size is 64 K.

AcceptAllContent

Specifies whether users can include content in POST, PUT, GET, and HEAD requests when a Content-Length or Transfer-encoding header is contained in the request header. You can specify one of the following values for this attribute:
- `True` if content is expected and read for all requests
- `False` if content is only expected and read for POST and PUT requests.

The default value is `True`.

ChunkedResponse

Specifies whether the plug-in must use chunks the response to the client when a Transfer-Encoding: Chunked response header is present in the response.

This attribute applies to the IIS, Oracle iPlanet, and Lotus Domino web servers only. The IBM HTTP Server automatically handles the chunking of the response to the client.

You can specify one of the following values for this attribute:
- `true` if the plug-in is to chunk the response to the client when a Transfer-Encoding: Chunked response header is present in the response.
- `false` if the response is not to be chunked.

The default value is `false`.

IISPluginPriority

Specifies the priority in which the IIS web server loads the WebSphere web server plug-in. Specify one of the following values for this attribute:
- `High`
- `Medium`
- `Low`

The default value is `High`.

Note: The IIS web server uses this value during startup. Therefore, you must restart the web server before this change takes effect.

Note: The default value of High ensures that all requests are handled by the WebSphere web server plug-in before they are handled by any other filter or extensions. If problems occur when using a priority of Medium or Low, you must rearrange the order or change the priority of the interfering filter or extension.

TrustedProxyEnable

Permits the web server plug-in to interface with the proxy servers and load balancers that are listed for the TrustedProxyList custom property. When this property is set to `true`, the proxy servers and load balancers in this trusted proxy list can set values for the $WSRA and $WSRH internal headers. The $WSRA internal header is the IP address of the remote host, which is typically the browser, or an internal address that is obtained by Network Address Translation (N.A.T.). The $WSRH internal header is the host name of the remote host. This header information enables the web server plug-in to interface with that specific proxy server or load balancer.

When you use this custom property you must also use the TrustedProxyList custom property to specify a list of trusted proxy servers and load balancers. Also, you must clear the Remove special headers check box on the Request Routing panel within the administrative console. For more information, see the documentation on web server plug-in request routing properties.

TrustedProxyList

Specifies a comma delimited list of all proxy servers or load balancers that have permission to interface with this web server plug-in. You must use this property with the `TrustedProxyEnable=true` custom property setting. If the TrustedProxyEnable custom property is set to `false`, this list is ignored.

SSLConsolidate

Specifies whether the web server plug-in is to compare the setup of each new SSL transport with the setup of other SSL transports that are already defined in the configuration file. When you set this property to `true`, and the plug-in determines that the keyring and CertLabel values specified for the new SSL transport match the values specified for an already defined SSL transport, the plug-in uses the existing SSL environment instead of creating a new SSL environment. Creating fewer SSL environments means that the plug-in requires less memory, and the plug-in initialization time decreases, thereby optimizing your overall GSkit environment.

SSLPKCSDriver

Specifies the fully qualified name of the loadable module that interfaces with an optional SSL co-processor. The fully qualified name must include the directory path and the module name.

SSLPKCSPassword

Specifies the password for the SSL co-processor with which the module, specified for the SSLPKCSDriver custom property, is interfacing.

If you are using an IBM HTTP Server, you can use the sslstash program to create a file that contains this password. In this situation, you can specify the fully-qualified name of that file, instead of the actual password, as the value for this custom property.

Log

Describes the location and level of log messages that are written by the plug-in. If a log element is not specified within the configuration file, then, in some cases, log messages are written to the web server error log.

For example, you might specify the following line of code:

```
<Log LogLevel="Error" Name="/opt/WebSphere/AppServer60/logs/http_plugin.log"/>
```

- **Name**

  Specifies the fully qualified path to the log file to which the plug-in writes error messages. Specify exactly one attribute for each log.

  If the file does not exist, then one is created. If the file exists, then it is opened in append mode, and the previous plug-in log messages remain.
- **LogLevel**

  Specifies the level of detail of the log messages that the plug-in writes to the log. Specify zero or one of the following values for each log.

Log Level Value	Log Level Description
Trace	All of the steps in the request process are logged in detail.
Stats	The server selected for each request and other load balancing information relating to request handling is logged.
Warn	All warning and error messages resulting from abnormal request processing are logged
Error	Only error messages resulting from abnormal request processing are logged
Debug	All of the critical steps performed in processing requests are logged.
Detail	All of the information about requests and responses are logged.

If a LogLevel value is not specified for the Log element, the default value, Error, is used.

Note: Be careful when setting the level to Trace. Multiple messages are logged at this level, which can consume disk space quickly. Do not use a Trace setting in a normally functioning environment because it adversely affects performance.

Property Name="esiEnable" Value="*true/false*"

Enables or disables the Edge Side Include (ESI) processor. If the ESI processor is disabled, the other ESI elements in this file are ignored.

You can set `Value` to `true` or `false`. By default, the ESI processor is enabled with its value set to `true`.

Property Name="esiMaxCacheSize" Value="*integer*"

Specifies, in 1 KB units, the maximum size of the cache. The default maximum size of the cache is 1024 KB (1 MB). If the cache is full, the first entry deleted from the cache is the entry that is closest its expiration time.

Property Name="ESIInvalidationMonitor" Value="*true/false*"

Specifies whether the ESI processor receives invalidations from the application server.

You can set `Value` to `true` or `false`. By default, this property is set to `false`.

Property Name="FIPSEnable" Value="*true/false*"

Specifies whether the Federal Information Processing Standard (FIPS) is enabled for making Secure Sockets Layer (SSL) connections to the application server. Set this property to `true`, if FIPS is enabled on the application server.

You can set `Value` value to `true` or `false`. By default, this property is set to `false`.

Property Name="PluginInstallRoot" Value="*C:\IBM\WebSphere\Plugins*"

Specifies the installation path for the plug-in. This property is mandatory if using the Global Security Kit (GSKit) because WebSphere Application Server supports the local installation of the GSKit instead of a global installation. The attribute value is set to a fully qualified path to the plug-in installation root.

Supported names recognized by the transport are keyring, stashfile, and password. By default, this property is set to `none`.

ServerCluster

Specifies a group of servers that are generally configured to service the same types of requests. Specify one or more clusters for each configuration.

In the simplest case, the cluster contains only one server definition. In the case in which more than one server is defined, the plug-in load balances across the defined servers by using either a Round Robin or a Random algorithm. The default algorithm is Round Robin.

The following code is an example of a ServerCluster element.

```
<ServerCluster Name="Servers">
<ClusterAddress Name="ClusterAddr">
<Transport Hostname="192.168.1.2" Port="9080" Protocol="HTTP"/>
<Transport Hostname="192.168.1.2" Port="9443" Protocol="HTTPS">
<Property Name="Keyring" value="c:/WebSphere/AppServer/keys/keyring.kdb"/>
<Property Name="Stashfile" value="c:/WebSphere/AppServer/keys/keyring.sth"/>
</Transport>
</ClusterAddress>
<Server Name="Server1">
<Transport Hostname="192.168.1.3" Port="9080" Protocol="HTTP"/>
<Transport Hostname="192.168.1.3" Port="9443" Protocol="HTTPS">
<Property Name="Keyring" value="c:/WebSphere/AppServer/keys/keyring.kdb"/>
```

```
<Property Name="Stashfile" value="c:/WebSphere/AppServer/keys/keyring.sth"/>
</Transport>
</Server>
<Server Name="Server2">
<Transport Hostname="192.168.1.4" Port="9080" Protocol="HTTP"/>
<Transport Hostname="192.168.1.4" Port="9443" Protocol="HTTPS">
<Property Name="Keyring" value="c:/WebSphere/AppServer/keys/keyring.kdb"/>
<Property Name="Stashfile" value="c:/WebSphere/AppServer/keys/keyring.sth"/>
</Transport>
</Server>
<Server Name="Server3">
<Transport Hostname="192.168.1.5" Port="9080" Protocol="HTTP"/>
<Transport Hostname="192.168.1.5" Port="9443" Protocol="HTTPS">
<Property Name="Keyring" value="c:/WebSphere/AppServer/keys/keyring.kdb"/>
<Property Name="Stashfile" value="c:/WebSphere/AppServer/keys/keyring.sth"/>
</Transport>
</Server>
<PrimaryServers>
<Server Name="Server1"/>
<Server Name="Server2"/>
</PrimaryServers>
<BackupServers>
<Server Name="Server3"/>
</BackupServers>
</ServerCluster>
```

The z/OS PTF UK35083 package includes the SSL interface change for the z/OS HTTP Server, Version 5.3, that corresponds to this web server plug-in change. Therefore, you must apply this PTF to your system before the new web server plug-in SSL interface can function properly.

| You must also include the `SSLMODE=MULTI` option in the `httpd.conf` file for the IBM HTTP Server for z/OS,
| Version 5.3. The SSLMODE=ON option is not supported in Version 7.0 or higher.

If the `SSLMode multi` option is not specified in the `httpd.conf` file, or if you do not have the z/OS PTF UK35083 package applied to your system, you might receive error message IMW0584W. This message indicates that the SSL mode, which is specified for the HTTP Server, is not compatible with the SSL mode for the web server plug-in that is used with the IBM HTTP Server for z/OS, Version 5.3. In either of these situations, unpredictable results might occur.

For the web server plug-ins for both the IBM HTTP Server for z/OS, Version 5.3 and the IBM HTTP Server on z/OS powered by Apache:

- If you use a `kdb` file with a stashfile in the hierarchical file system (HFS), specify both the `Property Name=keyring` and the `Property Name=stashfile` elements, as shown in the preceding example.

| **gotcha:** The format of the values you specify for these elements is different from what you specified in
| earlier versions of the product.

- If you use a System Authorization Facility (SAF) keyring, instead of a `kdb` file, you must create the following two custom plug-in properties from the administrative console:

  KeyringLocation
  Specify the directory location of the SAF keyring as the value for this property. When you save this configuration change, this directory location becomes the value of the keyring property in the `plugin-cfg.xml` file.

  StashfileLocation
  Specify "" (null) as the value for this property. When you save this configuration change, "" (null) becomes the value of the stashfile property in the `plugin-cfg.xml` file

  See "Web server plug-in configuration properties" on page 178 for instructions on how to create `KeyringLocation` and `StashfileLocation` from the administrative console.

  Use the following example for the SAF keyring:

```
<Transport Hostname="appserver.example.com" Port="9443" Protocol="https">
<Property name="keyring" value="safkeyring:///SAF_keyring_name"/>
<Property Name="stashfile" value=""/>
</Transport>
```

- **Name**

  Specifies the logical or administrative name to be used for this group of servers. Specify one attribute for each ServerCluster.

- **LoadBalance**

  The following values can be specified for this attribute:

  Round Robin

  Random

  The Round Robin implementation has a random starting point. the first application server is picked randomly. Round Robin is then used to pick application servers from that point forward. This implementation ensures that in multiple process based wed servers, all of the processes don't start up by sending the first request to the same Application Server.

  The Random implementation also has a random starting point. However with this implementation all subsequent servers are also randomly selected. Therefore, the same server might get selected repeatedly while other servers remain idle.

  The default load balancing type is Round Robin.

- **IgnoreAffinityRequests**

  Specifies whether the plug-in ignores the number of affinity requests made to a server when selecting servers based on the Round Robin algorithm. Specify zero or one attribute for each ServerCluster. The value is `true` or `false`. If the value is set to `false`, the number of affinity requests made is also taken into account in the server selection process.

  The default value is `true`, which means the number of affinity requests made are not used in the Round Robin algorithm.

- **RetryInterval**

  Specifies an integer value for the length of time that elapses from the time that a server is marked down to the time that the plug-in tries a connection again. The default is 60 seconds. Specify zero or one attribute for each ServerCluster.

- **RemoveSpecialHeaders**

  The plug-in adds special headers to the request before it is forwarded to the application server. These headers store information about the request that is used by the application. By default, the plug-in removes these headers from incoming requests before adding the headers it is supposed to add. Specify zero or one attribute for each ServerCluster.

  The value can be `true` or `false`. Setting the attribute to false introduces a potential security exposure by not removing headers from incoming requests.

- **CloneSeparatorChange**

  Tells the plug-in to expect the plus character (+) as the clone separator. Some pervasive devices cannot handle the colon character (:) that is used to separate clone IDs in conjunction with session affinity. You must change application server configurations so that an application server separates clone IDs with the plus character as well. Specify zero or one attribute for each ServerCluster.

  The value can be `true` or `false`.

- **PostSizeLimit**

  The maximum number of KBs (1024 byte) blocks of request content allowed for the plug-in to attempt to send the request to an application server. If a request is received that is greater than this size, the plug-in fails the request. The default value is -1 byte, which indicates that there is no limit for the post size. Specify zero or one attribute for each ServerCluster.

- **PostBufferSize**

Specifies, in KBs, the maximum buffer size that is used when the content of an HTTP request is read. If the application server that initially receives a request cannot process that request, the data contained in this buffer is sent to another application server. It then attempts to have that application server process the request.

This option improves the availability of the plug-in. Pending requests with content are tried again if the selected application server is not responding. If the value is set to zero, the requests with content are not buffered and are not tried again. The default value is 64. Specify zero or one attribute for each ServerCluster.

- **Server**

  Specifies a WebSphere Application Server instance that is configured to handle requests routed to it, based on the routing rules of the plug-in configuration. The server corresponds to an application server running on either the local machine or a remote machine. Specify zero or one attribute for each ServerCluster.

  - **Name**

    Specifies the administrative or logical name for the server. Specify exactly one attribute for each Server.

  - **CloneID**

    If this unique ID is present in the HTTP cookie header of a request, or the URL if using URL rewriting, the plug-in routes the request to this particular server, provided all other routing rules are met. If a CloneID is not specified in the server, then session affinity is not enabled for this server. There can be zero or one attribute for each Server.

    This attribute is used with session affinity. When this attribute is set, the plug-in checks the incoming cookie header or URL for **JSESSIONID**. If **JSESSIONID** is found, then the plug-in looks for one or more clone IDs. If clone IDs are found, and a match is made to the value specified for this attribute, then the request is sent to this server rather than load balanced across the cluster.

    Note: If you are not using session affinity, then remove these clone IDs from the configuration because there is added request processing in the plug-in when these values are set. If clone IDs are not in the plug-in, then it is assumed that session affinity is not enabled, and the request is load balanced across the cluster.

  - **WaitForContinue**

    Specifies whether to use the HTTP 1.1 `100 Continue` support before sending the request content to the application server. Possible attribute values are `true` or `false`. The default value is false; the plug-in does not wait for the `100 Continue` response from the application server before sending the request content because it is a performance hit. Specify zero or one attribute for each Server.

    This property is ignored for POST requests to prevent a failure from occurring if the application server closes a connection because of a keep-alive timeout.

    Enable this function true when configuring the plug-in to work with certain types of proxy firewalls.

  - **LoadBalanceWeight**

    Specifies the weight associated with this server when the plug-in performs weighted Round Robin load balancing. Specify zero or one attribute for each Server. The starting value for a server can be any integer between `0` and `20`. However, specify zero only for a server that is not running.

    The LoadBalanceWeight value for each server is decremented for each request that is processed by that server. After the weight for a particular server in a server cluster reaches zero, only requests with session affinity are routed to that server. When all servers in the cluster reach a weight of zero, the weights for all servers in the cluster are reset, and the algorithm restarts.

    Note: When a server is not running, set the weight for that server to zero. The plug-in can then reset the weights of the servers that are still running, and maintain proper load balancing.

  - **ConnectTimeout**

Enables the plug-in to perform non-blocking connections with the application server. Non-blocking connections are beneficial when the plug-in is unable to contact the destination to determine if the port is available or unavailable. Specify zero or one attribute for each Server.

If a ConnectTimeout value is not specified, the plug-in performs a blocking connect in which the plug-in sits until an operating system times out (as long as 2 minutes depending on the platform) and allows the plug-in to mark the server *unavailable*. A value of `0` causes the plug-in to perform a blocking connect. A value greater than 0 specifies the number of seconds you want the plug-in to wait for a successful connection. If a connection does not occur after that time interval, the plug-in marks the server *unavailable* and fails over to one of the other servers defined in the cluster. The default value is `5` seconds.

– **ExtendedHandshake**

Is used when a proxy firewall is between the plug-in and the application server. In such a case, the plug-in is not failing over, as expected. Specify zero or one attribute for each Server.

The plug-in marks a server as down when the connect() method fails. However, when a proxy firewall is in between the plug-in and the application server, the connect() method succeeds, even though the back-end application server is down. This causes the plug-in to not failover correctly to other application servers.

The plug-in performs some handshaking with the application server to ensure that it is started before sending the request. This scenario enables the plug-in to failover in the event the application server is down.

The value can be `true` or `false`.

– **MaxConnections**

Specifies the maximum number of pending connections to an application server that can be flowing through a web server process at any point in time. Specify one element for each Server.

For example, in the following scenario:
- The application server is fronted by five nodes that are running an IBM HTTP Server.
- Each node starts two processes.
- The MaxConnections attribute is set to 50.

In this example, the application server can potentially get up to 500 connections. Multiply the number of nodes, 5, by the number of processes, 2, and then multiply that number by the number specified for the MaxConnections attribute, 50, for a total of 500 connections.

By default, MaxConnections is set to -1. If this attribute is set to either zero or -1, there is no limit to the number of pending connections to the application servers.

– **Transport**

Specifies the transport for reading and writing requests to a particular WebSphere Application Server instance. The transport provides the information that is necessary to determine the location of the application server to which the request is sent. If the server has multiple transports that are defined to use the same protocol, the first one is used. Specify one or more elements for each Server.

It is possible to configure the server to have one non-secure transport and one that uses SSL. In this configuration, a match of the incoming request protocol is performed to determine the appropriate transport to use to send the request to the application server.

- **Hostname**

  Specifies the host name or IP address of the machine on which the WebSphere Application Server instance is running. There is exactly one attribute for each transport.

- **Port**

  Specifies the port on which the WebSphere Application Server instance is listening. There is exactly one attribute for each transport.

- **Protocol**

  Specifies the protocol to use when communicating over this transport -- either HTTP or HTTPS. There is exactly one attribute for each transport.

– **Property**

Specify zero, one, or more elements for each transport. When the protocol of the transport is set to HTTPS, use this element to supply the various initialization parameters, such as password, keyring and stashfile. For example, the portion of the `plugin-cfg.xml` file containing these elements might look like the following code:

```
<Transport Hostname="192.168.1.2" Port="9443" Protocol="HTTPS">
<Property Name="keyring" value="c:/WebSphere/AppServer/keys/keyring.kdb"/>
<Property Name="stashfile" value="c:/WebSphere/AppServer/keys/keyring.sth"/>
<Property Name="password" value="WebAS"/>
```

- **Name**

  Specifies the name of the property that is being defined. Supported names recognized by the transport are keyring, stashfile, and password.

  Note: The only name that can be specified for the WebSphere HTTP plug-in for z/OS is password. If you specify keyring and stashfile, they are ignored.

  Specify exactly one attribute for each Property.

- **Value**

  Specifies the value of the property being defined. Specify exactly one attribute for each property.

– **ServerIOTimeout**

Enables the plug-in to set a timeout value, in seconds, for sending requests to and reading responses from the application server.

If you set the ServerIOTimeout attribute to a positive value, this attempt to contact the server ends when the timeout occurs. However, the server is not considered unavailable and future requests are still sent to the server on which the unavailable timeout occurred.

If you set the ServerIOTimeout attribute to a negative value, the server is considered unavailable whenever a timeout occurs, and no future requests are sent to the server on which the timeout occurred.

If a value is not set for the ServerIOTimeout attribute, the plug-in, by default, uses blocked I/O to write request to and read response from the application server until the TCP connection times out. For example, you might specify the following setting:

```
<Server Name="server1" ServerIOTimeout=300>
```

In this situation, if an application server stops responding to requests, the plug-in waits 300 seconds (5 minutes) before timing out the TCP connection. Setting the ServerIOTimeout attribute to a reasonable value enables the plug-in to timeout the connection sooner, and transfer requests to another application server when possible.

When selecting a value for this attribute, remember that sometimes it might take several minutes for an application server to process a request. Setting the value of the ServerIOTimeout attribute too low might cause the plug-in to send a false server error response to the client.

| The default value is `900`, which is equivalent to 15 minutes.

Note: The ServerIOTimeout limits the amount of time the plug-in waits for each individual read or write operation to return. ServerIOTimeout does not represent a timeout for the overall request.

For additional recommendations on how to configure the ServerIOTimeout attribute, see the web server plug-in configuration technote on the IBM Support website.

- **ClusterAddress**

  A ClusterAddress is like a server element in that you can specify the same attributes and elements as for a server element. The difference is that you can define only one of them within a ServerCluster. Use a ClusterAddress when you do not want the plug-in to perform any type of load balancing because you already have some type of load balancer in between the plug-in and the application server.

Note: If you include a ClusterAddress tag, you must include the `Name` attribute on that tag. The plug-in uses the `Name` attribute to associate the cluster address with the correct host and port. If you do not specify the Name attribute, the plug-in assigns the cluster address the name that is specified for the server that is using the same host and port.

```
<ClusterAddress Name="MyClusterAddr">
<Transport Hostname="192.168.1.2" Port="9080" Protocol="HTTP"/>
<Transport Hostname="192.168.1.2" Port="9443" Protocol="HTTPS">
</ClusterAddress>
```

If a request comes in that does not have affinity established, the plug-in routes it to the cluster address, if defined. If affinity has been established, then the plug-in routes the request directly to the clone, bypassing the cluster address entirely. If no cluster address is defined for the server cluster, then the plug-in load balances across the servers in the primary servers list.

There can be zero or one element for each ServerCluster.

- **PrimaryServers**

  Specifies a list of servers to which the plug-in routes requests for this cluster. If a list of primary servers is not specified, the plug-in routes requests to servers defined for the server cluster. Specify zero or one element for each ServerCluster.

- **BackupServers**

  Specifies a list of servers to which requests are sent if all servers that are specified in the primary servers list are unavailable. The plug-in does not load balance across the backup servers, but traverses the list in order until no servers are left in the list or until a request is successfully sent and a response is received from an application server. Specify zero or one element for each ServerCluster.

VirtualHostGroup

Specifies a group of virtual host names that are specified in the HTTP Host header. Use this property to group virtual host definitions together that are configured to handle similar types of requests.

The following example shows a VirtualHostGroup element and associated elements and attributes:

```
<VirtualHostGroup Name="Hosts">
<VirtualHost Name="www.x.com"/>
<VirtualHost Name="www.x.com:443"/>
<VirtualHost Name="*:8080"/>
<VirtualHost Name="www.x.com:*"/>
<VirtualHost Name="*:*"/>
</VirtualHostGroup>
```

- **Name**

  Specifies the logical or administrative name to be used for this group of virtual hosts. Specify exactly one attribute for each VirtualHostGroup.

- **VirtualHost**

  Specifies the name used for a virtual or real machine used to determine if incoming requests must be handled by WebSphere Application Server. Use this element to specify host names that are in the HTTP Host header which must be seen for requests that need to be handled by the application server. You can specify specific host names and ports for incoming requests or specify an asterisk (*) for either the host name, port, or both.

  There can be one or more elements for each VirtualHostGroup.

  – **Name**

    Specifies that the name in the HTTP Host header that matches the name in the VirtualHost. Specify exactly one attribute for each VirtualHost.

    The value is a host name or IP address and port combination, separated by a colon.

    You can configure the plug-in to route requests to the application server based on the incoming HTTP Host header and port for the request. The Name attribute specifies those combinations.

You can use a wildcard for this attribute. The only acceptable solutions are either an asterisk (*) for the host name, an asterisk for the port, or an asterisk for both. An asterisk for both means that any request matches this rule. If no port is specified in the definition, the default HTTP port of 80 is assumed.

UriGroup

Specifies a group of URIs that are specified on the HTTP request line. The same application server must be able to handle the URIs. The route compares the incoming URI with the URIs in the group to determine if the application server handles the request.

The following example shows a UriGroup element and associated elements and attributes:

```
<UriGroup Name="Uris">
<Uri Name="/servlet/snoop"/>
<Uri Name="/webapp/*"/>
<Uri Name="*.jsp"/>
</UriGroup>
```

- **Name**

  Specifies the logical or administrative name for this group of URIs. Specify exactly one attribute for each UriGroup.
- **Uri**

  Specifies the virtual path to the resource that is serviced by WebSphere Application Server. Each URI specifies the incoming URLs that the application server needs to handle. You can use a wildcard in these definitions. There can be one or more attributes for each UriGroup.
  - **Name**

    Specifies the actual string to specify in the HTTP request line to match successfully with this URI. You can use a wildcard within the URI definition. You can specify rules such as *.jsp or /servlet/* to be handled by WebSphere Application Server. When you assemble your application, if you specify **File Serving Enabled**, then only a wildcard URI is generated for the web application, regardless of any explicit servlet mappings. If you specify **Serve servlets by classname**, then the following URI is generated: `<Uri Name="`*`Web_application_URI`*`/servlet/*">`

    There is exactly one attribute for each URI.
  - **AffinityCookie**

    Specifies the name of the cookie that the plug-in uses when trying to determine if the inbound request has session affinity. The default value is **JSESSIONID**.

    See the description of the CloneID attribute for additional session affinity information.

    There can be zero or one attribute for each URI.
  - **AffinityURLIdentifier**

    Specifies the name of the identifier that the plug-in uses when trying to determine if the inbound request has affinity specified in the URL to a particular clone. The default value is **jsessionid**.

    See the description of the CloneID attribute for additional session affinity information.

    There can be zero or one attribute for each URI.

Route

Specifies a request routing rule by which the plug-in determines if an incoming request must be handled by WebSphere Application Server.

The route definition is the central element of the plug-in configuration. It specifies how the plug-in handles requests, based on certain characteristics of the request. The route definition contains the other main elements: a required ServerCluster, and either a VirtualHostGroup, UriGroup, or both.

Using the information that is defined in the VirtualHostGroup and the UriGroup for the route, the plug-in determines if the incoming request to the web server is sent on to the ServerCluster element that is defined in this route.

See the following example of this element:

```
<Route VirtualHostGroup="Hosts" UriGroup="Uris" ServerCluster="servers/>
```

- **VirtualHostGroup**

  Specifies the group of virtual hosts that are used in route determination. The incoming host header and server port are matched to determine if this request is handled by the application server.

  It is possible to omit this property from the route definition. If it is not present, then every request matches during the virtual host match portion of route determination.

  There can be zero or one attribute for each Route.
- **UriGroup**

  Specifies the group of URIs to use for determining the route. Select zero or one group for each route. The incoming URI for the request is matched to the defined URIs in this group to determine whether this request is handled by the application server.

  It is possible to omit this property from the route definition. If it is not present, then every request matches during the URI match portion of route determination.
- **ServerCluster**

  Specifies the cluster that receives the requests that successfully matches the route. Select exactly one cluster for each route.

  The cluster is used to handle this request. If both the URI and the virtual host matching is successful for this route, then the request is sent to one of the servers that is defined within this cluster.

RequestMetrics

Used to determine whether request metrics are enabled, and how to filter the requests based on the Internet Protocol (IP) and Uniform Resource Identifiers (URI) filters when request metrics are enabled.

See the following example of this element:

```
<RequestMetrics armEnabled="false"  loggingEnabled="true"
   rmEnabled="false" traceLevel="PERF_DEBUG">
```

- **armEnabled**

  Specifies whether the ARM 4 agent is enabled in the plug-in. When it is set to `true`, the ARM 4 agent is called.

  Note: For the SunOne (iPlanet) web Server the following directive must be included in the `obj.conf` file to enable ARM 4 support:

  ```
  AddLog fn="as_term"
  ```

  If this directive is not included, the arm_stop procedure is never called.

  Select zero or one attribute for RequestMetrics
- **loggingEnabled**

  Specifies whether request metrics logging is enabled in the plug-in. When it is set to `true` and the traceLevel is not set to NONE, the request response time, and other request information, is logged. When it is set to `false`, there is no request logging. The value of loggingEnabled depends on the value specified for the system property, com.ibm.websphere.pmi.reqmetrics.loggingEnabled. When this system property is not present, loggingEnable is set to `true`. Specify exactly one attribute for RequestMetrics.
- **rmEnabled**

  Specifies whether the request metrics are enabled in the plug-in. When it is set to `true`, the plug-in, request metrics, inspects the filters and logs the request trace record in the plug-in log file. This action is performed if a request passes the filters. When this attribute is set to `false`, the rest of the request metrics attributes are ignored. Specify exactly one attribute for RequestMetrics.

- **traceLevel**

  Indicates how much information is logged when the `rmEnabled` attribute is set to `true`. When this attribute is set to `NONE`, no request logging is performed. When this attribute is not set to `NONE`, and loggingEnabled is set to `true`, the request response time, and other request information, is logged when the request is done. Specify exactly one attribute for RequestMetrics.

- **filters**

  When rmEnabled is `true`, the filters control which requests are traced. Specify zero, one, or two attributes for RequestMetrics.

  – **enable**

    When enable is `true`, the type of filter is on and requests must pass the filter. Specify exactly one attribute for each filter.

  – **type**

    There are two types of filters: SOURCE_IP (for example, client IP address) and URI. For the SOURCE_IP filter type, requests are filtered based on a known IP address. You can specify a mask for an IP address using the asterisk (*). If the asterisk is used, the asterisk must always be the last character of the mask, for example 127.0.0.*, 127.0.*, 127*. For performance reasons, the pattern matches character by character, until either an asterisk is found in the filter, a mismatch occurs, or the filters are found as an exact match.

    For the URI filter type, requests are filtered based on the URI of the incoming HTTP request. The rules for pattern matching are the same as matching SOURCE_IP address filters.

    If both URI and client IP address filters are enabled, request metrics requires a match for both filter types. If neither is enabled, all requests are considered a match.

    There is exactly one attribute for each filter.

  – **filterValues**

    Specifies the detailed filter information. Specify one or multiple attributes for each filter.

    - **value**

      Specifies the filter value for the corresponding filter type. This value might be either a client IP address or a URI. Specify exactly one attribute for each filterValue.

  – **enableESIToPassCookies**

    Specifies whether to allow forwarding of session cookies to WebSphere Application Server when processing ESI include requests. If the value is set to `true`, this custom property is enabled. If the value is set to `false`, the custom property is disabled. By default, the value is set to `false`.

  – **GetDWLMTable**

    Specifies whether to allow a newly created plug-in process to proactively request a partition table from WebSphere Application Server before it handles any HTTP requests. This custom property is used only when memory-to-memory session management is configured. If the value is set to `true`, this custom property is enabled. If the value is set to `false`, the custom property is disabled. By default, the value is set to `false`.

Web server plug-in custom properties

If you are using a web server plug-in, you can add one or more of the following custom properties to the configuration settings for that plug-in.

Complete these steps to add a web server plug-ins custom property.

1. In the administrative console, select **Servers > Server Types > Web servers >** *web_server_name* **> Plug-in properties > Custom properties > New**.

2. Under **General Properties**, specify the name of the custom property in the **Name** field and a value for this property in the **Value** field. You can also specify a description of this property in the **Description** field.
3. Click **Apply** or **OK**.
4. Click **Save** to save your configuration changes.
5. Re-generate and propagate the `plugin-cfg.xml` file.

Note: You can update the *global* `plugin-cfg.xml` file using the administrative console or running the GenPluginCfg command for all of the clusters in a cell. However, you must delete the `config/cells/plugin-cfg.xml` file before you update the *global* `plugin-cfg.xml` file. If you do not delete the `config/cells/plugin-cfg.xml` file, only the new properties and their values are added to the *global* `plugin-cfg.xml` file. Any updates to existing plug-in property values are not added to the *global* `plugin-cfg.xml` file.

CertLabel

Specifies the label of the certificate within the keyring that the plug-in is to use when the web container requests a client certificate from the plug-in. This custom property does not apply to any client certificate that is coming from the SSL connection with the browser. If you are using an SSL co-processor, and the plug-in is not running on a z/OS or IBM i system, if you specify the token label, followed by a colon, as the value for this custom property the entire CertLabel value is used as the keyring label.

gotcha: You can only use this custom property if you are running on Version 7.0.0.3 or later.

Data type	Boolean
Default	False

GetDWLMTable

Specifies whether the plug-in should prefetch the partition table. When this custom property is enabled, the plugin prefetches the partition table so that affinity requests are maintained. The GetDWLMTable custom property must be enabled when memory-to-memory session management is configured for WebSphere Application Server.

Data type	String
Default	False

HTTPMaxHeaders

Specifies the maximum number of headers that can be included in a request or response that passes through the web server plug-in. If a request or response contains more than the allowable number of headers, the web server plug-in drops the extra headers.

Data type	Integer
Range	1 - 4000
Default	300

If you prefer, instead of adding this custom property, you can manually add the following values to the plugin-cfg.xml file:

```
HTTPMaxHeaders = "<value>" in the Config tag. Example :
<Config ASDisableNagle="false" AcceptAllContent="false"
AppServerPortPreference="HostHeader" ChunkedResponse="false"
```

```
FIPSEnable="false" HTTPMaxHeaders="2500"
IISDisableNagle="false" IISPluginPriority="High"
IgnoreDNSFailures="false" RefreshInterval="60"
ResponseChunkSize="64" VHostMatchingCompat="false">
```

SSLConsolidate

Specifies whether the web server plug-in is to compare the setup of each new SSL transport with the setup of other SSL transports that are already defined in the configuration file. When you set this property to `true`, and the plug-in determines that the keyring and CertLabel values specified for the new SSL transport match the values specified for an already defined SSL transport, the plug-in uses the existing SSL environment instead of creating a new SSL environment. Creating fewer SSL environments means that the plug-in requires less memory, and the plug-in initialization time decreases, thereby optimizing your overall GSkit environment.

Data type	Boolean
Default	False

SSLPKCSDriver

Specifies the fully qualified name of the loadable module that interfaces with an optional SSL co-processor. The fully qualified name must include the directory path and the module name.

Data type	String
Default	None

SSLPKCSPassword

Specifies the password for the SSL co-processor with which the module, specified for the SSLPKCSDriver custom property, is interfacing.

If you are using an IBM HTTP Server, you can use the sslstash program to create a file that contains this password. In this situation, you can specify the fully-qualified name of that file, instead of the actual password, as the value for this custom property.

Data type	String
Default	None

TrustedProxyEnable

Permits the web server plug-in to interface with the proxy servers and load balancers that are listed for the TrustedProxyList custom property. When this property is set to `true`, the proxy servers and load balancers in this trusted proxy list can set values for the $WSRA and $WSRH internal headers. The $WSRA internal header is the IP address of the remote host, which is typically the browser, or an internal address that is obtained by Network Address Translation (N.A.T.). The $WSRH internal header is the host name of the remote host. This header information enables the web server plug-in to interface with that specific proxy server or load balancer.

When you use this custom property you must also use the TrustedProxyList custom property to specify a list of trusted proxy servers and load balancers. Also, you must clear the Remove special headers check box on the Request Routing panel within the administrative console. For more information, see the documentation on web server plug-in request routing properties.

Data type	Boolean
Default	False

TrustedProxyList

Specifies a comma delimited list of all proxy servers or load balancers that have permission to interface with this web server plug-in. You must use this property with the `TrustedProxyEnable=true` custom property setting. If the TrustedProxyEnable custom property is set to `false`, this list is ignored.

Example:

```
TrustedProxyList = myProxyServer1.myDomain.com,myProxyServer2.com,192.168.0.1
```

Data type	String
Default	None

Web server plug-in configuration properties

Web server plug-in configuration properties are set on various panels of the administrative console. The table provided indicates on which panel a particular property is set.

Table 20. Web server plug-in configuration properties. The following table indicates which panel in the administrative console you need to use to manually configure a web server plug-in property.

Administrative console panel	Field name	Configuration property name
In the administrative console, click **Servers** > **Server Types** > **Web servers** > *web_server_name* > **Plug-in properties**	Refresh configuration interval	RefreshInterval
In the administrative console, click **Servers** > **Server Types** > **Web servers** > *web_server_name* > **Plug-in properties**	Plug-in log file name	Log->name
In the administrative console, click **Servers** > **Server Types** > **Web servers** > *web_server_name* > **Plug-in properties**	Plug-in logging	Log->LogLevel
In the administrative console, click **Servers** > **Server Types** > **Web servers** > *web_server_name* > **Plug-in properties**	Ignore DNS failures during web server startup	IgnoreDNSFailures
In the administrative console, click **Servers** > **Server Types** > **Web servers** > *web_server_name* > **Plug-in properties**	KeyringLocation	Keyring
In the administrative console, click **Servers** > **Server Types** > **Web servers** > *web_server_name* > **Plug-in properties**	StashfileLocation	Stashfile
In the administrative console, click **Servers** > **Server Types** > **Web servers** > *web_server_name* > **Plug-in properties** > **Custom properties** > **New**	FIPSEnable	FIPSEnable

Table 20. Web server plug-in configuration properties (continued). The following table indicates which panel in the administrative console you need to use to manually configure a web server plug-in property.

In the administrative console, click **Servers** > **Web Servers** > *Web_server_name*> **Plug-in properties** > **Custom properties** > **New**	TrustedProxyEnable	TrustedProxyEnable
In the administrative console, click **Servers** > **Web Servers** > *Web_server_name*> **Plug-in properties** > **Custom properties** > **New**	TrustedProxyList	TrustedProxyList
In the administrative console, click **Servers** > **Web Servers** > *Web_server_name*> **Plug-in properties** > **Custom properties** > **New**	SSLConsolidate	SSLConsolidate
In the administrative console, click **Servers** > **Web Servers** > *Web_server_name*> **Plug-in properties** > **Custom properties** > **New**	SSLPKCSDriver	SSLPKCSDriver
In the administrative console, click **Servers** > **Web Servers** > *Web_server_name*> **Plug-in properties** > **Custom properties** > **New**	SSLPKCSPassword	SSLPKCSPassword
In the administrative console, click **Servers** > **Server Types** > **Web servers** > *Web_server_name* > **Plug-in properties** > **Request routing**	Load balancing option	LoadBalance
In the administrative console, click **Servers** > **Server Types** > **Web servers** > *web_server_name* > **Plug-in properties** > **Request Routing**	Clone separator change	CloneSeparatorChange
In the administrative console, click **Servers** > **Server Types** > **Web servers** > *web_server_name* > **Plug-in properties** > **Request Routing**	Retry interval	RetryInterval
In the administrative console, click **Servers** > **Server Types** > **Web servers** > *web_server_name* > **Plug-in properties** > **Request routing**	Maximum size of request content	PostSizeLimit
In the administrative console, click **Servers** > **Server Types** > **Web servers** > *web_server_name* > **Plug-in properties** > **Request routing**	Size of the buffer that is used to cache POST requests	PostBufferSize

Table 20. Web server plug-in configuration properties (continued). The following table indicates which panel in the administrative console you need to use to manually configure a web server plug-in property.

In the administrative console, click **Servers** > **Server Types** > **Web servers** > *web_server_name* > **Plug-in properties** > **Request routing**	Remove special headers	RemoveSpecialHeaders
In the administrative console, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Web server plug-in properties**	Server role	PrimaryServers and BackupServers list
In the administrative console, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Web server plug-in properties**	Connect timeout	Server ConnectTimeout
In the administrative console, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name*, and then, in the Additional Properties section, click **Web server plug-in properties**.	The read and write timeouts for all the connections to the application server	ServerIOTimeout
In the administrative console, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Web server plug-in properties**	Use extended handshake to check whether Application Server is running	Server Extended Handshake
In the administrative console, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Web server plug-in properties**	Send the header "100 Continue" before sending the request content	WaitForContinue
In the administrative console, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Web server plug-in properties**	Maximum number of connections that can be handled by the Application Server	Server MaxConnections
In the administrative console, click **Servers** > **Server Types** > **Web servers** > *web_server_name* > **Plug-in properties** > **Request and Response**	Application server port preference	AppServerPortPreference
In the administrative console, click **Servers** > **Web Servers** > *Web_server_name* > **Plug-in properties** > **Request and Response**	Enable Nagle algorithm for connections to the Application Server	ASDisableNagle
In the administrative console, click **Servers** > **Server Types** > **Web servers** > *web_server_name* > **Plug-in properties** > **Request and Response**	Enable Nagle Algorithm for the IIS Web Server	IISDisableNagle

Table 20. Web server plug-in configuration properties (continued). The following table indicates which panel in the administrative console you need to use to manually configure a web server plug-in property.

In the administrative console, click **Servers** > **Server Types** > **Web servers** > *web_server_name* > **Plug-in properties** > **Request and Response**	Virtual host matching	VHostMatchingCompat
In the administrative console, click **Servers** > **Server Types** > **Web servers** > *web_server_name* > **Plug-in properties** > **Request and Response**	Maximum chunk size used when reading the response body	ResponseChunkSize
In the administrative console, click **Servers** > **Server Types** > **Web servers** > *web_server_name* > **Plug-in properties** > **Request and Response**	Accept content for all requests	AcceptAllContent
In the administrative console, click **Servers** > **Server Types** > **Web servers** > *web_server_name* > **Plug-in properties** > **Request and Response**	Chunk HTTP response to the client	ChunkedResponse
In the administrative console, click **Servers** > **Server Types** > **Web servers** > *web_server_name* > **Plug-in properties** > **Request and Response**	Priority used by the IIS Web Server when loading the plug-in configuration file	IISPluginPriority
In the administrative console, click **Servers** > **Server Types** > **Web servers** > *web_server_name* > **Plug-in properties** > **Caching**	Enable Edge Side Include (ESI) processing to cache the responses	ESIEnable
In the administrative console, click **Servers** > **Server Types** > **Web servers** > *web_server_name* > **Plug-in properties** > **Caching**	Maximum cache size	ESIMaxCacheSize
In the administrative console, click **Servers** > **Server Types** > **Web servers** > *web_server_name* > **Plug-in properties** > **Caching**	Enable invalidation monitor to receive notifications	ESIInvalidationMonitor
In the administrative console, click **Servers** > **Web Servers** > *Web_server_name* > **Plug-in properties** > **Caching**	Forwarding of session cookies to WebSphere Application Server.	enableESIToPassCookies

Web server plug-in tuning tips

Important tips for web server plug-in tuning include how to balance workload and improve performance in a high stress environment. Balancing workloads among application servers in a network fronted by a web server plug-in helps improve request response time.

Balancing workloads

During normal operation, the backlog of connections pending to an application server is bound to grow. Therefore, balancing workloads among application servers in a network fronted by a web server plug-in helps improve request response time.

You can limit the number of connections that can be handled by an applications server. To do this:

1. Go to the **Servers > Server Types > WebSphere application servers >** *server_name*.
2. In the Additional Properties section, click **Web Server Plug-in properties** .
3. Select **Use maximum number of connections** for the Maximum number of connections that can be handled by the Application server field.
4. Specify in the Connections field the maximum number of connections that you want to allow.
5. Then click **Apply** and **Save**.

When this maximum number of connections is reached, the plug-in, when establishing connections, automatically skips that application server, and tries the next available application server. If no application servers are available, an HTTP 503 response code will be returned to the client. This code indicates that the server is currently unable to handle the request because it is experiencing a temporary overloading or because maintenance is being performed.

The capacity of the application servers in the network determines the value you specify for the maximum number of connections. The ideal scenario is for all of the application servers in the network to be optimally utilized. For example, if you have the following environment:

- There are 10 application servers in a cluster.
- All of these application servers host the same applications (that is, Application_1 and Application_2).
- This cluster of application servers is fronted by five IBM HTTP Servers.
- The IBM HTTP Servers get requests through a load balancer.
- Application_1 takes approximately 60 seconds to respond to a request
- Application_2 takes approximately 1 second to respond to a request.

Depending on the request arrival pattern, all requests to Application_1 might be forwarded to two of the application servers, say Appsvr_1 and Appsvr_2. If the arrival rate is faster than the processing rate, the number of pending requests to Appsvr_1 and Appsvr_2 can grow.

Eventually, Appsvr_1 and Appsvr_2 are busy and are not able to respond to future requests. It usually takes a long time to recover from this overloaded situation.

If you want to maintain 2500 connections, and optimally utilize the Application Servers in this example, set the number of maximum connections allowed to 50. (This value is arrived at by dividing the number of connections by the result of multiplying the number of Application Servers by the number of web servers; in this example, 2500/(10x5)=50.)

Limiting the number of connections that can be established with an application server works best for web servers that follow use a single, multithreaded process for serving requests.

Windows IBM HTTP Server uses a single, multithreaded process for serving requests. No configuration changes are required.

AIX HP-UX Solaris IBM HTTP Server typically uses multiple multithreaded processes for serving requests. Specify the following values for the properties in the web server configuration file (httpd.conf) to prevent the IBM HTTP Server from using more than one process for serving requests.

```
ServerLimit          1
ThreadLimit          1024
StartServers         1
MaxClients           1024
MinSpareThreads      1
MaxSpareThreads      1024
ThreadsPerChild      1024
MaxRequestsPerChild  0
```

Improving performance in a high stress environment

Windows If you use the default settings for a Microsoft Windows operating system, you might encounter web server plug-in performance problems if you are running in a high stress environment. To avoid these problems, consider tuning the TCP/IP setting for this operating system. Two of the keys setting to tune are TcpTimedWaitDelay and MaxUserPort.

To tune the TcpTimedWaitDelay setting, change the value of the tcp_time_wait_interval parameter from the default value of 240 seconds, to 30 seconds:

1. Locate in the Windows Registry:
   ```
   HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\tcpip\Parameters\TcpTimedWaitDelay
   ```
   If this entry does not exist in your Windows Registry, create it by editing this entry as a new DWORD item.
2. Specify, in seconds, a value between 30 and 300 inclusive for this entry. (It is recommended that you specify a value of 30.)

To tune the MaxUserPort setting:

1. Locate in the Windows Registry:
   ```
   HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\tcpip\Parameters\MaxUserPort
   ```
   If this entry does not exist in your Windows Registry, create it by editing this entry as a new DWORD item.
2. Set the maximum number of ports to a value between 5000 and 65534 ports, inclusive. (It is recommended that you specify a value of 65534,)

See the Microsoft website for more information about these settings.

Chapter 3. Using the DataPower appliance manager

The DataPower® appliance manager automatically starts if you issue a request to the DataPower appliance manager and it is not already started. You can initiate a request using the wsadmin tool, or by selecting any of the administrative console pages that enable you to view or change settings for DataPower appliances, firmware, or managed sets, or the administrative console page that is used to monitor DataPower appliance manager tasks. The appliance manager also automatically starts when the deployment manager starts if there are any DataPower appliances configured in the appliance manager.

Before you begin

The first time that you use the DataPower appliance manager, you must add at least one appliance to the appliance manager. Before adding an appliance to the appliance manager, you must verify that:

- The appliance that you are adding is at a Version 3.6.0.4 or higher firmware level. The appliance manager cannot manage an appliances that is not at a Version 3.6.0.4 or higher firmware level.
- The appliance manager can communicate with the port that is used for the XML Management interface AMP endpoint on the appliance. The appliance manager uses this port to send commands to an appliance.
- The appliance can communicate with the port that is used for the DataPowerMgr_inbound_secure endpoint on the deployment manager. The appliance uses this port to send events to the appliance manager.
- The Appliance Management Protocol (AMP) endpoint is enabled for each appliance. If the XML Management interface AMP endpoint was disabled during installation, use the DataPower WebGUI to enable the AMP endpoint.
- There is not a firewall between the deployment manager and the appliances that will be part of a managed set. If there is a firewall between the deployment manager and the appliances, that firewall might prevent the appliance manager from communicating with the appliances in a managed set.

gotcha: Remember that the XML management interface port, which defaults to port number 5550, is different from the Web Management Service port, which defaults to port number 9090. The DataPower appliance manager uses the XML management interface port to manage the DataPower appliances. You use the Web Management Service port to access the WebGUI on the DataPower appliance. You can use the same user ID and password to access both the XML management interface and the WebGUI.

About this task

When the DataPower appliance manager starts, it automatically starts the channel chain which causes a bind to the port that is defined for the DataPower appliance manager.

To access the DataPower appliance manager, in the administrative console, click **Servers > DataPower**, and then perform one or more of the following actions. As previously stated, if this is the first time that you are using the DataPower appliance manager, you must add at least one appliance to the appliance manager before completing any of the other actions.

If you need to shut down the DataPower appliance manager, in the administrative console, click **Servers > DataPower > Appliance manager**, and then click **Shutdown Manager**.

Procedure

- Add an appliance to the DataPower appliance manager.
- Add a new firmware version to the DataPower appliance manager.
- Create a new managed set.

- Monitor the tasks that are queued for the DataPower appliance manager
- Modify appliance manager settings.
- Manage the DataPower appliance domins.
- View the managed domains history.
- Manage versions of sharable appliance settings.

What to do next

You can configure Secure Sockets Layer (SSL) for the DataPower appliance manager.

WebSphere DataPower appliance manager overview

WebSphere DataPower appliance manager provides a set of capabilities for managing sets of appliances. DataPower appliance manager can be used to manage appliances with a 3.6.0.4 or higher level of firmware.

IBM® WebSphere® DataPower SOA Appliances are purpose-built, easy-to-deploy network devices that simplify, help secure, and accelerate your XML and Web services deployments.

The first time you use DataPower appliance manager, you must add appliances, firmware versions, and managed sets. Verify that each appliance that you want to add has a 3.6.0.4 or higher level of firmware. Also verify that the Appliance Management Protocol (AMP) endpoint is enabled for each appliance. If the XML Management interface AMP endpoint was disabled during installation, use the DataPower WebGUI to enable the AMP endpoint.

For security reasons, the DataPower appliance manager does not include Crypto material, such as keys and certificates, in the shareable settings and domain versions that it creates. Therefore, after you add or replace an appliance, you must manually add any Crypto material that you want to apply for that appliance.

gotcha: The DataPower WebGUI is different from the WebSphere Application Server administrative console that you use to administer the DataPower appliance manager. The DataPower WebGUI is a separate user interface on the DataPower appliance that is used to configure the appliance.

Managed sets

A managed set is a collection of appliances that share the same hardware type, model type, and feature license set. A managed set synchronizes sharable appliance settings, managed domains, and firmware across multiple appliances.

A managed set can contain one or more appliances. An appliance is not actively managed unless it is a member of a managed set. You must first add an appliance to the DataPower appliance manager, and then add the appliance to a managed set.

Sharable appliance settings

Sharable appliance settings are the global attributes for an appliance that can be shared with other appliances. For example, NTP configuration and SNMP configuration are sharable appliance settings, but appliance-specific settings, such as IP address and role-based management attributes are not sharable appliance settings,

Sharable appliance settings are not managed until an appliance is added to a managed set. After you add an appliance to a managed set, any changes that you make to the sharable appliance settings, using the DataPower WebGUI or command line interface, are synchronized from the master appliance to all of the subordinate appliances in the managed set.

Master appliances

The master appliance is the appliance in the managed set that is used to synchronize sharable appliance settings and managed domains for all appliances within the managed set. Each managed set must have at least one master appliance. Each managed set might also have subordinate appliances.

All subordinate appliances are synchronized with the master appliance, and have the same sharable appliance settings and managed domains as the master appliance. You use the DataPower WebGUI or command line interface to change the sharable appliance settings, or a managed domain on a master appliance. The DataPower command line interface is a command line user interface on the DataPower appliance that can be used to configure the appliance.

Sharable appliance settings and managed domains on subordinate appliances are automatically overwritten whenever a change is made to the master appliance. If you use the DataPower WebGUI or the DataPower command line interface to change the sharable appliance settings, or a managed domain on a master appliance, the appliance manager detects the change, and propagates the changes to the remaining appliances in the managed set. Therefore, if the sharable appliance settings or a managed domain is changed on a subordinate appliance, making the sharable appliance settings or a managed domain different from what is on the master appliance, the appliance manager automatically overwrites the changes on the subordinate appliance with the sharable appliance settings, or managed domain values that are on the master appliance.

gotcha: Ensure that any changes that you make to the shareable appliance settings or a managed domain on a master appliance can be used for all of the appliances in the managed set.

Managed domains

DataPower supports the use of application domains to partition configuration information into self contained units that are easier to manage. Because an application domain consists of resources that are configured to provide and support one or more services, you can use domains to group configuration information on a appliance. For example, you might set up a domain for a set of business applications because you want to keep their DataPower appliance configuration separate from the DataPower appliance configuration for the other applications on that appliance.

A managed domain is a domain on the master appliance that has been added to a managed set in the DataPower appliance manager. The DataPower appliance manager uses the managed domain to synchronize configuration changes to the subordinate appliances that are part of the managed set.

Both master appliances and subordinate appliances can also have unmanaged domains. The DataPower appliance manager does not make configuration changes to unmanaged domains.

gotcha: The DataPower appliance manager synchronizes managed domains from the master appliance to the subordinate appliances in the managed set. However, it is possible that the managed domain might not be completely functional on all of the subordinate appliances. For example, the managed domain might not be completely functional on a subordinate appliance if a service object, such as an XML firewall, in the managed domain has a listening port conflict on that subordinate appliance.

Versions of sharable appliance settings

Whenever the appliance manager detects that you have used the DataPower WebGUI or DataPower command line interface to change the sharable appliance settings for a master appliance, the appliance manager automatically creates a new version of the sharable appliance settings. This new version of the sharable appliance settings is called a settings version. The newest settings version is, by default, the active version for the managed set. This new settings version is automatically copied to all of the appliances in the managed set.

You can deploy any version of the sharable appliance settings to a managed set. Whenever you deploy a settings version, the deployed version becomes the active version until the sharable appliance settings are changed, or you deploy a different settings version. If you have more than one version of sharable appliance settings for a managed set, you can complete these tasks.

gotcha: Changes to sharable appliance settings only apply for appliances that are members of the same managed set. Changes are not propagated to appliances that are members of a different managed set.

- Copy a version of sharable appliance settings to another managed set. The sharable appliance settings are applied to all appliances in this other managed set.

  gotcha: After the initial copy of the sharable appliance settings, the two managed sets are managed independently. Therefore, future changes to the sharable appliance settings in one managed set are not reflected in the other managed set.
- Delete an inactive version of sharable appliance settings. You cannot delete an active version. You can also specify the maximum number of versions that you want to keep.
- Deploy a version of the sharable appliance settings. You deploy a version of the sharable appliance settings to make a different version active. When the different version becomes the active version, that version is deployed to all of the members of the managed set.

Versions of managed domains

When you change a managed domain on a master appliance, the appliance manager automatically detects the change and creates a new version of the managed domain. The newest version of the managed domain is, by default, the active version for the managed set. This new version of the domain is automatically copied to all appliances in the managed set. You can deploy any version of a managed domain to a managed set, and that deployed version automatically becomes the active managed domain for that managed set.

When a managed domain is deleted from a master appliance, the domain is automatically recreated on the master appliance. To delete a managed domain, you must convert the managed domain to an unmanaged domain.

When you have multiple versions of a managed domain, you can perform the following tasks:

- Copy a version of the managed domain to another managed set. The domain is applied to all of the appliances in the managed set.

  gotcha: After the initial copy of the managed domain, the two managed sets are managed independently. Therefore, future changes to the sharable appliance settings in one managed set are not reflected in the other managed set.
- Delete an inactive version of the managed domain. You cannot delete an active version. You can also specify the maximum number of versions that you want to keep.
- Deploy a version of the managed domain. You deploy a version of a managed domain to make that version the active version. The active version is then deployed to all members of the managed set.

Firmware

Firmware version files must be obtained from the IBM support website and are specific to appliance types, model types, and licensed features. When a firmware version is loaded to an appliance, it must be compatible with the appliance type, model type, and licensed features. DataPower appliance manager manages appliances with a 3.6.0.4 or higher level of firmware. A firmware file is typically in a scrypt2 format.

Versions of firmware

The appliance manager automatically determines the firmware version, intended model type, appliance type, and licensed features provided by libraries in the firmware. The appliance manager allows the firmware types to be deployed only to matching appliances.

A firmware version must exist in the DataPower appliance manager before that version can be deployed to appliances. If the firmware version that is running on an appliance is not in this file, a managed set that includes that appliance can only contain that single appliance, because the appliance manager cannot deploy that firmware version to any other appliance.

When you deploy a particular version of firmware, that version becomes the active version. When you have more than one version of firmware, you can perform the following tasks:

- Deploy a version firmware to the managed set. You deploy a version of firmware to roll back, or upgrade the firmware on the appliances to a specific version. Whenever a new version is deployed, that version becomes the active version for the managed set, and is deployed to all of the appliances in that managed set.

  The firmware versions, that are in the DataPower appliance manager, can be used with multiple managed sets if the appliance type and model type are the same and the licensed features are compatible.
- Delete a version of firmware. You cannot delete an active version of firware. As an alternative to deleting firmware versions, you can configure the maximum number of versions of any one object that you want to keep.

gotcha: Do not use the DataPower 3.6.0.28, 3.6.0.29, or 3.6.0.30 level of firmware for a managed set. These firmware levels might cause the DataPower appliance manager to unnecessarily create new shareable appliance settings versions, or domain versions, and then synchronize these new versions across the managed set.

Setting up and administering a managed set

If you want to create at least one managed set, you must complete the following tasks. These tasks make it possible for the DataPower appliance manager to manage the appliances in a managed set:

- Add one or more DataPower appliances to the appliance manager.
- Create the firmware version in the appliance manager that you want used on all of the appliances in the managed set. You can have different firmware versions for different managed sets, or you can share the firmware versions between managed sets.
- Create a managed set for all of the appliances that are intended to share the same firmware version, shared appliance settings and managed domains.

After at least one managed set is created, you can complete the following tasks in any order:

- Add appliances to a managed set.

  Note: Do not add the same DataPower appliance to a managed set in two different DataPower appliance managers. If a DataPower appliance manager discovers that another DataPower appliance manager is managing an appliance, the discovering DataPower appliance manager removes that appliance from its managed set. If this appliance is the only appliance in that managed set, the discovering DataPower appliance manager also removes all of the shareable settings and domain versions that are associated with that managed set. When this situation occurs, you cannot recover any historical versions of the shareable appliance settings and domains that do not exist on the other DataPower appliance manager.

  For example, if you create the following managed sets:

  – Managed set MS1 on DataPower appliance manager A that contains DataPower appliance X.

- – Managed set MS2 on DataPower appliance manager B that also contains DataPower appliance X.

  When DataPower appliance manager A discovers that DataPower appliance manager B is also managing DataPower appliance called X, DataPower appliance manager A issues an error message to the deployment manager log file, that indicates that the appliance is being managed by DataPower appliance manager B, and removes appliance X from the managed set MS1. Because appliance X was the only managed appliance in MS1, DataPower appliance manager A removes all of the shareable appliance settings and domain versions that are associated with MS1. You will not be able to recover any historical versions that existed on DataPower appliance manager A, but do not exist on DataPower appliance manager B.
- Manage versions of the firmware, sharable appliance settings, and managed domains with roll back capability.
- Monitor appliance synchronization and operation status.

You can also use the administrative console to manage long running tasks for the DataPower appliance manager, view the status of these tasks, or delete one or more of these task. However, you cannot delete a task to stop the task from being completed. The only way to interrupt a running task, or prevent the appliance manager from running a task, is to shutdown the appliance manager. Shutting down the appliance manager terminates all running and queued tasks.

Propagating sharable appliance settings and managed domains from master to non-master appliances

If there are multiple appliances in the managed set, then the changes made to the active version of the sharable appliance settings are propagated to the subordinate appliances in the managed set. Likewise, changes made to the managed domains of master appliances are propagated to the subordinate appliances in the managed set.

The appliance manager also detects when subordinate appliances are available. For example, if sharable appliance settings are changed for the master appliance, but the subordinate appliances are not available, then the master appliance and the subordinate appliances cannot be synchronized. When the subordinate appliances are available, the appliance manager detects the change in status and initiates synchronization from the master appliance to the subordinate appliances in the managed set.

Adding DataPower appliances to the DataPower appliance manager

You can use the DataPower appliance manager that is provided with the product to administer a DataPower appliance. After you add an appliance to the DataPower appliance manager, you can make it part of a managed set of appliances if you want the DataPower appliance manager to keep the shared appliance settings for this appliance synchronized with the shared appliance settings of the other appliances that are part of that managed set.

Before you begin

- Verify that the appliance that you are adding is at a Version 3.6.0.4 or higher firmware level. The appliance manager cannot manage an appliances that is not at a Version 3.6.0.4 or higher firmware level.
- Verify that the appliance manager can communicate with the port that is used for the XML Management interface AMP endpoint on the appliance. The appliance manager uses this port to send commands to an appliance.
- Verify that the appliance can communicate with the port that is used for the DataPowerMgr_inbound_secure endpoint on the deployment manager. The appliance uses this port to send events to the appliance manager.

- Verify that the Appliance Management Protocol (AMP) endpoint is enabled for each appliance. If the XML Management interface AMP endpoint was disabled during installation, use the DataPower WebGUI to enable the AMP endpoint.
- Verify that there is not a firewall between the deployment manager and the appliances that will be part of a managed set. If there is a firewall between the deployment manager and the appliances, that firewall might prevent the appliance manager from communicating with the appliances in a managed set.

About this task

You can use the administrative console to add a new DataPower appliance or to access the DataPower WebGUI. If you want to access the DataPower WebGUI, in the administrative console, click **Servers > DataPower > Appliances**, and then click **Launch**. Complete the following steps if you want to add a add a new DataPower appliance.

gotcha: Remember that the XML management interface port, which defaults to port number 5550, is different from the Web Management Service port, which defaults to port number 9090. The DataPower appliance manager uses the XML management interface port to manage the DataPower appliances. You use the Web Management Service port to access the WebGUI on the DataPower appliance. You can use the same user ID and password to access both the XML management interface and the WebGUI.

Procedure

1. In the administrative console, click **Servers > DataPower > Appliances > New**.
2. Specify a unique name for the appliance in the **Name** field. An appliance name must be unique and cannot contain an invalid character. The name field cannot contain the characters # $ @ \ / , : ; " * ? < > | = + & % or '.
3. Specify an IP address or fully qualified host name in the **Host name** field.
4. Specify the XML management interface port that the DataPower appliance uses in the **Administrative port** field.
5. Specify the ID that you want to use to connect to the DataPower appliance in the **User ID** field.
6. Specify the password that you want to associate with the user ID in the **Password** and **Verify password** fields.
7. Click **OK** to add the new appliance.

Results

The DataPower appliance manager is managing this appliance.

What to do next

You can change appliance settings, add the appliance to a managed set, or remove the appliance from the DataPower appliance manager.

gotcha: For security reasons, the DataPower appliance manager does not include Crypto material, such as keys and certificates, in the shareable settings and domain versions that it creates. Therefore, after you add an appliance to the appliance manager, you must manually add any Crypto material that you want to apply for the new appliance.

Modifying DataPower appliance settings

You can use the administrative console to modify the shared appliance settings for an appliance.

Before you begin

Verify that the DataPower appliance manager is managing this appliance. You cannot use the administrative console to change the sharable appliance settings for an appliance that has not been added to the DataPower appliance manager.

About this task

Complete the one or more of the following actions to make modifications to the settings for an appliance.

Procedure

1. From the administrative console, click **Servers > DataPower > Appliances >** *appliance_name*.
2. Specify a different IP address or fully qualified host name in the **Host name** field.
3. Specify a different XML management interface port for the DataPower appliance to use in the **Administrative port** field.

   gotcha: Remember that the XML management interface port, which defaults to port number 5550, is different from the Web Management Service port, which defaults to port number 9090. The DataPower appliance manager uses the XML management interface port to manage the DataPower appliances. You use the Web Management Service port to access the WebGUI on the DataPower appliance. You can use the same user ID and password to access both the XML management interface and the WebGUI.
4. Specify a different user ID in the **User ID** field.
5. Specify a new password to associated with the user ID in the **Password** and **Verify password** fields.
6. Select an appliance type from the list of appliance types in the **Appliance type** field.
7. In the Domains section, expand **Managed Domains**, or **Unmanaged Domains** to view a list of managed and unmanaged domains that are associated with this appliance.

   If you want to change one or more managed domains to unmanaged domains, click **Servers > DataPower > Managed sets >** *managedset_name*, select the domains that you want to change, and then click**Unmanage**. Similarly, if you want to change one or more unmanaged domains to managed domains, select the domains that you want to change, and then click **Manage**.

   Clicking either **Manage** or **Unmanage**, creates an appliance manager task. If you want to know when this task completes, you can click **Servers > DataPower > Tasks** to check the status of this task.
8. Click **OK** to save a copy of your changed settings as a new settings version.

Removing a DataPower appliance

You can use the administrative console to remove an appliance from the DataPower appliance manager.

About this task

If you no longer want the DataPower appliance manager to manage one of you appliances, complete the following steps to remove that appliance from the DataPower appliance manager.

gotcha: Do not remove a DataPower appliance from a managed set if it is the only appliance in that managed set, until you have added the replacement appliance to that managed set. Removing the last appliance from a managed set causes the managed set to be deleted, and all of the versions of the shareable settings and domains to be removed from the DataPower appliance manager. There is not any way for you to recover these removed versions.

Procedure

1. In the administrative console, click **Servers > DataPower > Appliances**.
2. Verify that the appliance that you want to remove is not part of a managed set.

You cannot remove an appliance from the DataPower appliance manager if that appliance is part of a managed set. If the appliance that you want to remove has the value unmanaged in the Managed Set column, the appliance is not part of a managed set.

If the name of a managed set is specified in the Managed Set column, you must remove the appliance from the managed set before you remove it from the DataPower appliance manager.

- If the appliance that you want to remove from a managed set is the last appliance in that managed set, you must delete the managed set before you can remove the appliance from the DataPower appliance manager. When you delete the managed set, the appliance automatically becomes an unmanaged appliance and can be removed from the DataPower appliance manager.
- If the appliance that you want to remove from a managed set is not the last appliance in that managed set, complete the following actions to remove the appliance from that managed set.

a. Click Managed sets, and then click the name of the managed set that includes the appliance that you want to remove.

b. Click **Edit Membership**, select the appliance that you want to remove from the DataPower appliance manager, and then click **Remove**.

c. Click **Save** to save your changes.

3. Select the appliance that you want to remove from the DataPower appliance manager.

   If you want to remove multiple appliances from the DataPower appliance manager, you can select all of the appliances that you want to remove.
4. Click **Remove**.

Appliance collection

Use this page to add, change, or remove a DataPower appliance in the DataPower appliance manager. You can also use this page to view information about the DataPower appliances that are in the appliance manager. Adding appliances to a DataPower appliance manager enables you to automatically keep synchronized the settings for all of the appliances that are in a DataPower appliance manager managed set.

To view this administrative console page, click **Servers > DataPower > Appliances**.

To view the values specified for a DataPower appliance, click the name of that appliance name in the list of available appliances. The displayed appliance settings page shows the values specified.

To initiate an operation on a DataPower appliance, select the name of the appliance that you want to manage, and click the operation that you want performed on that appliance. Click **Launch** to access the DataPower WebGUI in the default domain on the selected appliance. Pop-ups must be enabled for your browser.

Name

Specifies the symbolic name of the DataPower appliance.

An appliance name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # $ @ \ /, : ; " * ? < > | = + & % '.

Host Name

Specifies the host name or IP address of the DataPower appliance.

Managed Set

Specifies the name of the managed set that the appliance is assigned to. Appliances that are assigned to a managed set must be removed from the managed set before they can be removed from the appliance manager.

Master

Specifies the master appliance for the managed set to which this appliance is a member. All appliances in the managed set are synchronized using the settings of the master appliance.

Operational Status

Specifies the operational status of the appliance. The status is always unknown if the appliance is not part of a managed set or if the managed set has no managed domains. If the appliance is part of a managed set, the status reflects the aggregated operational status of all of the managed domains on this appliance.

This field is read-only. The following values might appear in this field:

up Indicates that all of the service objects in all the managed domains on the appliance are enabled.

partial Indicates that the service objects in all the managed domains on the appliance are a mix of enabled, disabled and unknown.

down Indicates that all of the service objects in all the managed domains on the appliance are disabled.

unknown
Indicates that the state of all of the managed domains on the appliance could not be determined. It is possible that the state has yet to be retrieved, or that communication to the appliances has been lost. An appliance always has a status of unknown if the appliance is not part of a managed set, or if the appliance is part of a managed set that has no managed domains.

Synchronization Status

Specifies whether the appliances are synchronized with other appliances in the managed set.

This field is read-only. The following values might appear in this field:

synced
Indicates that the firmware, sharable appliance settings and managed domains for the appliance are synchronized with the active firmware, sharable appliance and domain versions for the managed set.

changes pending
Indicates that at least one firmware synchronization, sharable appliance settings synchronization, or managed domains synchronization is queued for the appliance.

in progress
Indicates that the synchronization is currently being processed. The appliance has at least one firmware synchronization, settings synchronization, or managed domain synchronization active. Other firmware synchronizations, settings synchronizations, or managed domain synchronizations might also be pending for this appliance.

unknown
Indicates that the synchronization status is not known for at least one firmware synchronization, settings synchronization, or managed domain synchronization. Typically, this status means that the appliance manager has not yet contacted the appliance to get the management status. Firmware synchronizations, settings synchronizations, or managed domain synchronizations might also be pending, or in progress, for this appliance.

error Indicates that an attempt to process a firmware synchronization, settings synchronization, or managed domain synchronization for this appliance has failed because of an error on the appliance. The appliance did respond, but responded with an error. This appliance might also have a management status of unknown, in progress or changes pending for a firmware synchronization, settings synchronization, or managed domain synchronization. See the error log for the deployment manager for more information.

unreachable
Indicates that an attempt to synchronize the appliance has failed because the appliance is not responding. This appliance might also have a management status of unknown, in progress,

changes pending or error for a firmware synchronization, settings synchronization, or managed domain synchronization. See the error log for the deployment manager for more information.

New appliance settings for the DataPower appliance

Use this page to specify the settings for an appliance that you are adding to the DataPower appliance manager.

To view this administrative console page, click **Servers > DataPower > Appliances > New**.

Name

Specifies a user-defined symbolic name for a DataPower appliance.

An appliance name must be unique, and cannot contain the following invalid characters: # $ @ \ / , : ; " * ? < > | = + & % '.

Host name

Specifies an IP address or the fully qualified host name of a DataPower appliance.

Administrative port

Specifies the XML management interface port used by the appliance.

User ID

Specifies the user ID used by DataPower appliance manager to log into an appliance.

Password

Specifies the password for the user ID provided.

Verify password

Also specifies the password for the user ID provided. If you change the password value, you must respecify the new value in this field to ensure that you entered it correctly.

Appliance settings

Use this page to change the settings for a DataPower appliance. Appliances are added to the DataPower appliance manager to synchronize the firmware, shareable appliance settings and managed domains of a group of appliances called a managed set.

To view this administrative console page, click **Servers > DataPower > Appliances >** *appliance_name*.

In the Domains section, you can expand **Managed domains** to view a list of the managed domains that are on the appliance, and the status of each of these domains. You can also expand **Unmanaged domains** to view the unmanaged domains that are on the appliance. Domains, by default, are not managed. Unmanaged domains are not copied to subordinate appliances in the managed set. Managed domains are copied to subordinate appliances in the managed set.

Name

Specifies a user-defined symbolic name for a DataPower appliance.

An appliance name must be unique, and cannot contain the following invalid characters: # $ @ \ / , : ; " * ? < > | = + & % '.

Host name

Specifies an IP address or the fully qualified host name of a DataPower appliance.

Administrative port

Specifies the XML management interface port used by the appliance.

User ID

Specifies the user ID used by DataPower appliance manager to login to an appliance.

Password

Specifies the password for the user ID provided.

Verify password

Also specifies the password for the user ID provided. If you change the password value, you must respecify the new value in this field to ensure that you entered it correctly.

Serial number

Specifies the serial number for the appliance. This field is read-only.

Appliance type

Specifies the appliance type, such as XA35, XS40, or XI50.

Licensed features

Specifies a comma-separated list of the licensed features that are on this appliance. All of the appliances in a managed set must have the same appliance type, model type and licensed features. Firmware has a list of the licensed features that are contained in the firmware. Any firmware that is loaded onto an appliance must have a list of licensed features that is compatible with the licensed features that are on that appliance.

Because managed sets contain appliances that all must be of the same appliance type, model type and licensed features, you cannot use a firmware with a managed set that has licensed features that are incompatible with the appliances in that managed set. If you attempt to load a firmware that has a list of features that is incompatible with the licensed features of the appliance, the firmware does not load, and an error message is issued.

Firmware level

Specifies the numeric level of the firmware that is currently on the appliance. Each firmware version in the DataPower appliance manager is for a specific appliance type, model type, list of licensed features, and level.

It is possible that a firmware version might already exist in the DataPower appliance manager that has the same appliance type, model type, list of licensed features, and level as the firmware that is currently on this appliance.

Firmware synchronization status

Specifies whether the firmware version for an appliance is synchronized with the active firmware version for the managed set. The status is unknown if the appliance is not part of a managed set. If the appliance is part of a managed set, the status indicates whether the firmware that is associated with the appliance is synchronized with the designated firmware for the managed set.

This field is read-only. The following values might appear in this field:

synched
: Indicates that the firmware version for this appliance is the same as the firmware version for the managed set.

changes pending
: Indicates that the firmware is being synchronized.

in progress
: Indicates that the synchronization for the firmware on this appliance is currently being processed.

unknown
: Indicates that the synchronization status for the firmware on this appliance is not known. Typically,

this status indicates that the appliance manager has not yet contacted the appliance to get the synchronization status, or the appliance is not a member of a managed set.

error Indicates that an attempt to synchronize the firmware that is associated with the managed set has failed because of an error on the appliance. The appliance did respond, but responded with an error. See the error log for the deployment manager for more information.

unreachable
Indicates that an attempt to synchronize the firmware on this appliance failed because the appliance is not responding. See the error log for the deployment manager for more information.

Settings synchronization status

Specifies whether the shareable appliance settings for an appliance are synchronized with the active settings version for the managed set. The status is unknown if the appliance is not part of a managed set. If the appliance is part of a managed set, the status indicates whether or not the sharable settings for the appliance are synchronized with the designated sharable appliance settings version for the managed set.

This field is read-only. The following values might appear in this field:

synched
Indicates that the sharable appliance settings for this appliance are the same as the sharable appliance settings version for the managed set.

changes pending
Indicates that the synchronization for the sharable appliance settings on this appliance is queued.

in progress
Indicates that the shareable appliance settings are being synchronized.

unknown
Indicates that the synchronization status for sharable appliance settings on this appliance is not known. Typically, this status indicates that the appliance manager has not yet contacted the appliance to get the synchronization status, or the appliance is not a member of a managed set, or the appliance is not a member of a managed set.

error Indicates that an attempt to synchronize the sharable appliance settings on this appliance failed because of an error. The appliance did respond, but responded with an error. See the error log for the deployment manager for more information.

unreachable
Indicates that an attempt to synchronize the sharable appliance settings on this appliance failed because the appliance is not responding. See the error log for the deployment manager for more information.

Model type

Specifies a numeric value for the appliance model. Some model types include 9001, 9002, and 9003. Each appliance is assigned a model type.

Synchronization status

Specifies the combined status of the firmware synchronization, the sharable settings synchronization, and the managed domains synchronization for the appliance.

This field is read-only. The following values might appear in this field:

synched
Indicates that the firmware, sharable appliance settings and managed domains for the appliance are synchronized with the active firmware, sharable appliance and domain versions for the managed set.

changes pending
Indicates that at least one firmware synchronization, sharable appliance settings synchronization, or managed domains synchronization is queued.

in progress
Indicates that the synchronization is currently being processed. The appliance has at least one firmware synchronization, settings synchronization, or managed domain synchronization active. Other firmware synchronizations, settings synchronizations, or managed domain synchronizations might also be pending for this appliance.

unknown
Indicates that the synchronization status is not known for at least one firmware synchronization, settings synchronization, or managed domain synchronization. Typically, this status means that the appliance manager has not yet contacted the appliance to get the management status. It is possible that other firmware synchronization, settings synchronization, or managed domain synchronizations are also pending or in progress for the appliance.

error Indicates that an attempt to process a firmware synchronization, settings synchronization, or managed domain synchronization for this appliance has failed because of an error. The appliance did respond, but responded with an error on the appliance. This appliance might also have a management status of unknown, in progress or changes pending for a firmware synchronization, settings synchronization, or managed domain synchronization. See the error log for the deployment manager for more information.

unreachable
Indicates that an attempt to synchronize the appliance has failed because the appliance is not responding. This appliance might also have a management status of unknown, in progress, changes pending or error for a firmware synchronization, settings synchronization, or managed domain synchronization. See the error log for the deployment manager for more information.

Name (in the Managed domains section)

Specifies the name of the managed domain.

Operational Status

Specifies the operational status of the managed domain. Each service inside of a DataPower domain on an appliance has an operational state of either enabled or disabled. The operational state of the domain in the DataPower appliance manager represents the aggregated operational status of all the service objects in the domain.

This field is read-only. The following values might appear in this field:

up Indicates that all of the service objects in the managed domain on the appliance are enabled.

partial Indicates that the service objects in the managed domain on the appliance are a mix of enabled, disabled and unknown.

down Indicates that all of the service objects in the managed domain on the appliance are disabled

unknown
Indicates that the state of the managed domain on the appliance cannot be determined. It is possible that the state has yet to be retrieved, or that communication to the appliance has been lost.

Synchronization status (in the Managed domains section)

Specifies the synchronization status of the managed domain on the appliance.

This field is read-only. The following values might appear in this field:

synched
Indicates that the managed domain on this appliance is synchronized with the domain version for the managed set.

changes pending
Indicates that the synchronization is queued.

in progress
Indicates that the synchronization is currently being processed.

unknown
Indicates that the synchronization status is not known. The managed domain synchronization state for the appliance is unknown. Typically, this status means that the appliance manager has not yet contacted the appliance to get the management status.

error Indicates that an attempt to synchronize the managed domain on the appliance has failed because of an error. The appliance did respond, but responded with an error. See the error log for the deployment manager for more information.

unreachable
Indicates that an attempt to synchronize the managed domain on the appliance has failed because the appliance is not responding. See the error log for the deployment manager for more information.

Name (in the Unmanaged domains section)

Specifies the name of any unmanaged domain on the device.

Replacing a DataPower appliance

If a problem occurs that makes a DataPower appliance unable to function properly, you can use the DataPower appliance manager to copy the configuration of that appliance to a different appliance.

Before you begin

Verify that the replacement DataPower appliance:

- Has the same device type, model type, and licensed features as the DataPower appliance that this appliance is replacing. The DataPower appliance manager requires all appliances in a managed set to be homogenous.
- Has been added to the DataPower appliance manager.

About this task

gotcha: Do not remove a DataPower appliance from a managed set if it is the only appliance in that managed set, until you have added the replacement appliance to that managed set. Removing the last appliance from a managed set causes the managed set to be deleted, and all of the versions of the shareable settings and domains to be removed from the DataPower appliance manager. There is not any way for you to recover these removed versions.

For security reasons, the DataPower appliance manager does not include Crypto material, such as keys and certificates, in the shareable settings and domain versions that it creates. Therefore, the following procedure does not include the restoration of Crypto material. After you replace an appliance, you must manually add the Crypto material for the replacement appliance.

Procedure

1. From the administrative console, click **Servers > DataPower > Managed sets >**
2. Click the name of the managed set to which the DataPower appliance that is being replaced belongs, and then click **Edit Membership**.
3. Select the replacement appliance, and then click **Add**
4. Click **Synchronize** to manually synchronize the firmware, shared appliance settings and managed domains throughout the managed set.

The appliance manager always attempts to keep the firmware, shared appliance settings and managed domains synchronized across all of the appliances in a managed set. However, when you can click **Synchronize**, you are forcing an immediate synchronization to occur.

5. Click **Edit Membership**, and select the appliance that you are replacing, and then click **Remove**.

Results

The replacement appliance is being added to the managed set, and the original appliance is being removed from the managed set.

What to do next

Click **Servers > DataPower > Tasks** to monitor the progress of the addition of the replacement appliance to the managed set, and the removal of the original appliance from the managed set.

Adding new firmware versions to the DataPower appliance manager

You can use the DataPower appliance manager to add a new firmware version to the DataPower appliance manager. Appliances that the DataPower appliance manager manages must have a 3.6.0.4 or higher firmware level.

Before you begin

Download one or more firmware versions from the IBM support site.

About this task

Complete the following steps if you want to use the administrative console to add a new firmware version.

Procedure

1. From the administrative console, click **Servers > DataPower > Firmware > New**.
2. Select either **Upload from local system**, or **Upload from remote system**.

   If you select **Upload from local system**, the firmware version is uploaded to the DataPower appliance manager, from your local file system.

   If you select **Upload from remote system**. the firmware version is uploaded to the DataPower appliance manager, from a remote file system.
3. Specify the fully qualified name of the file that you are uploading.

   If you do not know the fully qualified name of the file that you are uploading, you can use the browse function to search for the correct file.
4. Optional: Add a description of this firmware version in the **Comments** field.

   You might want to use this field to provide information that differentiates this firmware version from other firmware versions that are stored on the local or remote system.

   After you enter your comment, click **Apply**.
5. Click **Submit**.

What to do next

After a new firmware version is added, you can use the DataPower appliance manager to modify the settings for the new firmware version, add the new firmware version to a managed set, or delete the new firmware version when it is no longer needed.

Modifying settings for firmware versions

You can use the administrative console to add new firmware versions to the DataPower appliance manager, or to delete existing firmware versions. You can also use the administrative console to view the settings for the firmware versions that are on the DataPower appliance manager.

About this task

When you add a new firmware version to the DataPower appliance manager, you are not creating a new version. You are actually uploading a file that contains an existing firmware version, from either a remote or local file system, to the DataPower appliance manager. If you need to create a new firmware version, you must use the DataPower WebGUI. To access the DataPower WebGUI, in the administrative console, click **Servers > DataPower > Appliance**, and then click **Launch**.

To use the administrative console to view the settings for the current firmware version or to add or delete a firmware version, perform one or more of the following actions.

Procedure

1. To view the settings for the firmware versions on the DataPower appliance manager, click **Servers > DataPower > Firmware**.
2. To add a new firmware version to the DataPower appliance manager, click **New** After you select the file to upload, click **Submit**.
3. To delete firmware versions from the DataPower appliance manager, select the versions to delete, and then click **Delete**.
4. To add a comment about a firmware version that is on the DataPower appliance manager, click the name of that firmware version and add the comment in the **Comment** field.

   After you enter your comment, click **Apply**, and then click **Submit**.

Deleting firmware versions

You can delete a firmware version from the DataPower appliance manager, unless a managed set is using that firmware version. You cannot delete a firmware version that a managed set is using.

About this task

You can use the administrative console to delete firmware versions. Complete the following actions.

Procedure

1. In the administrative console, click **Servers > DataPower > Firmware**.
2. Select the firmware versions that you want to delete.
3. Click **Delete**.

Results

The firmware version is deleted from the DataPower appliance manager.

Firmware collection

Use this page to add a new firmware version to the appliance manager, view existing firmware versions, or delete a firmware version from the appliance manager. A firmware version must exist in the appliance manager before that version can be designated as the active firmware version for a managed set.

This page lists the firmware versions that can be used for appliances managed by DataPower appliance manager.

To view this administrative console page, click **Servers > DataPower > Firmware**.

To view the settings for a specific firmware version, click that firmware version in the list of available versions. The displayed firmware settings page shows the values that are specified for that firmware version. On the settings page, you can change the settings for the selected firmware version.

To add a new firmware version, click **New**.

To delete an existing firmware version, select the version that you want to delete and click **Delete**.

Version

Specifies the numeric version, or level, of the firmware. When combined with an appliance type, model type, and licensed features, the numeric version, or level, uniquely identifies a firmware version.

Appliance Type

Specifies the type of appliance that the firmware version is used for. Some appliance types include XA35, XS40, and XI50. The appliance type is based on the purpose for using the appliance.

Model Type

Specifies a numeric value for the appliance model. Some model types include 9001, 9002, and 9003.

Licensed Features

Specifies a comma-separated list of the licensed features that are on this appliance. All of the appliances in a managed set must have the same appliance type, model type and licensed features. Firmware has a list of the licensed features that are contained in the firmware. Any firmware that is loaded onto an appliance must have a list of licensed features that is compatible with the licensed features that are on that appliance.

Because managed sets contain appliances that all must be of the same appliance type, model type and licensed features, you cannot use a firmware with a managed set that has licensed features that are incompatible with the appliances in that managed set. If you attempt to load a firmware that has a list of features that is incompatible with the licensed features of the appliance, the firmware does not load, and an error message is issued.

Release Date

Specifies the date when the firmware was released by the manufacturer.

Firmware settings

Use this page to specify firmware settings for a firmware version in the DataPower appliance manager. A firmware version is a firmware image of a specific level that is used with a specific appliance type, model type, and set of licensed features.

To view this administrative console page, click **Servers > DataPower > Firmware >** *firmware_version*.

Version

Specifies the numeric version, or level, of the firmware. This field is read-only.

Release date

Specifies the date when the firmware was released by the manufacturer. This field is read-only.

Appliance type

Specifies the type of appliance that the firmware version is used for. Some appliance types include XA35, XS40, and XI50. This field is read-only.

Model type
Specifies a numeric value for the appliance model. Some model types include 9001, 9002, and 9003. This field is read-only.

Licensed features
Specifies a comma-separated list of the licensed features that are contained in this firmware. The firmware that is loaded onto an appliance must have a list of licensed features that is compatible with the licensed features of the appliance. This field is read-only.

All of the appliances that are part of a managed set must have the same appliance type, model type and licensed features. Therefore, you cannot use a firmware with a managed set if the firmware includes a list of licensed features that is not compatible with the appliances in the managed set. If you attempt to load firmware that includes a list of licensed features that is not compatible with the licensed features of the appliances in the managed set, the firmware does not load, and an error message is issued.

Comments
Specifies user-defined information about a firmware version.

Managed sets
Specifies a list of managed sets that use this firmware version. The firmware version must have the same appliance type, model type and licensed features as the appliances that are in the managed set. This field is read-only.

Managed set firmware settings

Use this page to change the firmware on a managed set. You can upload a file from a local or remote file system. The firmware for a managed set, must have the same value for appliance type, model type and licensed features as the devices in the managed set.

Note: Do not use the DataPower 3.6.0.28, 3.6.0.29, or 3.6.0.30 level of firmware for a managed set.

Typically, when you create a managed set of DataPower appliances, the DataPower appliance manager completes the following process:
- Creates a shareable appliance settings version from the shareable appliance settings on the master appliance.
- Synchronizes the settings across the managed set. This synchronization process is repeated whenever the shareable appliance settings change on the master appliance.

Similarly, when a domain is added to a managed set in the DataPower appliance manager, the DataPower appliance manager typically completes the following process:
- Creates a domain version from the domain on the master appliance.
- Synchronizes the domain version across the managed set. This synchronization process is repeated every time that the domain changes on the master appliance.

If you use the DataPower 3.6.0.28, 3.6.0.29, or 3.6.0.30 level of firmware for a managed set, the firmware level might cause the DataPower appliance manager to unnecessarily create new shareable appliance settings versions, or domain versions, and then synchronize these new versions across the managed set.

To view this administrative console page, click **Servers > DataPower > Managed sets >** *managed_set_name* > **Change firmware**.

From a version already in the DataPower appliance manager
Select this option if you want to use a firmware version that already exists in the appliance manager, and then select the version that you want to use from the list of available versions.

Upload from local system

Select this option if you want to upload a firmware version from a local file system. You can either type a fully qualified path name in the **Location** field, or you can select a file from the local file system.

Upload from remote system

Select this option if you want to upload a firmware version from a remote file system. You can either type a fully qualified path name in the **Location** field, or you can select a file from the remote file system.

Comments

Specifies user-defined information about a firmware version.

New firmware version settings

Use this page to upload a new firmware version from either a local or remote file system.

To view this administrative console page, click **Servers > DataPower > Managed sets > New**.

Upload from local system

Select this option if you want to upload a firmware version from a local file system. You can either type a fully qualified path name in the **Location** field, or you can select a file from the local file system.

Upload from remote system

Select this option if you want to upload a firmware version from a remote file system. You can either type a fully qualified path name in the **Location** field, or you can select a file from the remote file system.

Comments

Specifies user-defined information about a firmware version.

Adding a new managed set

Add a new managed set to synchronize settings for multiple appliances. A managed set is a grouping of appliances that share the same shareable appliance settings, managed domains and firmware version. Shareable appliance settings and managed domains are propagated to the subordinate appliances from the master appliance.

Before you begin

Verify the names of the DataPower appliances and firmware versions that you want to add to the new managed set.

About this task

Complete these steps to add a new managed set from the administrative console. You can click **Launch** to access the DataPower WebGUI to configure an appliance before you add the appliance to a managed set.

Procedure

1. From the administrative console, click **Servers > DataPower > Managed sets > New**. This starts the New managed set wizard.
2. Enter a unique name for the new managed set in the **Name** field. A managed set name must be unique and cannot contain an invalid character. The name field cannot contain the characters # $ @ \ / , : ; " * ? < > | = + & % or '.
3. Choose a master appliance, and then click **Next**.
4. Select appliances to add to the managed set, and click **Next**.
5. View the summary of actions, and then click **Finish**.

What to do next

Creating a new managed set might take several minutes to complete. When a new managed set is added, the DataPower appliance manager automatically creates a version of the sharable appliance settings from the master appliance and copies it to any subordinate appliances in the managed set. The firmware version for the managed set might also be automatically deployed to all of the appliances in the managed set that do not already have this firmware version.

After you create a managed set, you can use the administrative console to add, replace, or remove appliances from that managed set.

gotcha: Do not remove a DataPower appliance from a managed set if it is the only appliance in that managed set, until you have added the replacement appliance to that managed set. Removing the last appliance from a managed set causes the managed set to be deleted, and all of the versions of the shareable settings and domains to be removed from the DataPower appliance manager. There is not any way for you to recover these removed versions.

Modifying a managed set

You can change the members of a managed set, manually synchronize the sharable appliance settings, administer the versions of sharable appliance settings that are available for a managed set, change firmware versions, administer domains, or deploy a domain or sharable appliance settings version.

About this task

To modify a managed set, in the administrative console, click **Servers > DataPower > Managed sets >** *managedset_name*, and then complete one or more of the following actions.

Procedure

- To add appliances to the managed set, click **Edit Membership**, and then select the appliances that you want to add to this managed set.
- To remove appliances from the managed set, click **Edit Membership**, and then select the appliances that you want to remove from this managed set.
- To choose a different appliance as the master appliance, click **Edit Membership**, and then select a different appliance as the master appliance.
- Click **Synchronize** to manually synchronize the firmware, shared appliance settings and managed domains throughout the managed set.

  The appliance manager always attempts to keep the firmware, shared appliance settings and managed domains synchronized across all of the appliances in a managed set. However, you can click **Synchronize** if you need to manually force a synchronization to occur.
- To access the DataPower WebGUI to configure an appliance or a firmware version, click **Launch**.
- To modify which firmware version is the active firmware version for the managed set, in the Firmware section, click **Change firmware**.

  Note: Do not use the DataPower 3.6.0.28, 3.6.0.29, or 3.6.0.30 level of firmware for a managed set.

  Typically, when you create a managed set of DataPower appliances, the DataPower appliance manager completes the following process:

  – Creates a shareable appliance settings version from the shareable appliance settings on the master appliance.
  – Synchronizes the settings across the managed set. This synchronization process is repeated whenever the shareable appliance settings change on the master appliance.

  Similarly, when a domain is added to a managed set in the DataPower appliance manager, the DataPower appliance manager typically completes the following process:

  - Creates a domain version from the domain on the master appliance.
  - Synchronizes the domain version across the managed set. This synchronization process is repeated every time that the domain changes on the master appliance.

  If you use the DataPower 3.6.0.28, 3.6.0.29, or 3.6.0.30 level of firmware for a managed set, the firmware level might cause the DataPower appliance manager to unnecessarily create new shareable appliance settings versions, or domain versions, and then synchronize these new versions across the managed set.
- To view a list of settings versions that have been used for the managed set, in the Settings section, click **Version history**.

  You can then click **Detail**, if you want to change the version that is being used for this managed set, or if you want to delete a version.
- To view a list of appliances assigned to the managed set, click **Appliances**.
- To view a list of all of the managed domains on the master appliance, in the Managed domains section, expand the **Managed domains** field. Within this field, you can change a domain from managed to unmanaged, or access the DataPower WebGUI to configure a domain.
  1. To change a domain from managed to unmanaged, select the domain, and then click **Unmanage**.
  2. To access the DataPower WebGUI to configure a domain, click **Launch** at the top of the Managed Domains table.

     Clicking **Launch** launches the DataPower WebGUI in the managed domain on the master appliance.
- To view a list of all of the managed domains for this managed set, in the Managed domains section, expand the **Unmanaged domains on the master appliance** field. Within this field, you can change a domain from unmanaged to managed, or click **Launch** to access the DataPower WebGUI, and configure a domain.
  1. To change a domain from unmanaged to managed, select the domain, and then click **Manage**
  2. To access the DataPower WebGUI to configure a domain, click **Launch** at the top of the Managed Domains table.

What to do next

When you add appliances to or remove appliances from a managed set, perform a manual synchronization, change firmware versions, shareable settings or domain versions, an appliance manager task is created. To monitor the progress of these tasks, in the administrative console, click **Servers > DataPower > Tasks**.

Removing a managed set from the DataPower appliance manager

You can remove a managed set from the DataPower appliance manager if that managed set is not being used to manage appliances.

Before you begin

Verify that there are no appliances in the managed set that you want to remove from the DataPower appliance manager. If the managed set contains appliances, you must remove those appliances from that managed set before you try to remove the managed set from the appliance manager.

About this task

Complete the following actions to remove a managed set from the DataPower appliance manager.

Procedure

1. From the administrative console, click **Servers > DataPower > Managed sets**.
2. Select the managed set that you want to remove.

You can select multiple managed sets if there are multiple managed sets that you want to remove.

3. Click **Remove**.

Managed set collection for a DataPower appliance

Use this page to add, view, or delete a managed set. A managed set is a group of appliances whose firmware, shareable appliance settings and managed domains are all kept synchronized.

To view this administrative console page, click **Servers > DataPower > Managed sets**.

You can then:

- Click the name of one of the managed sets to view the values specified for that managed set configuration. The settings page displays and shows the current settings for the selected managed set.
- Click **New** if you want to create a new managed set.
- Select one of the managed sets in the list, and then click **Delete** if you want to delete that managed set.
- Click **Launch** to access the DataPower WebGUI in the default domain on the master appliance in the selected managed set. Pop-ups must be enabled. See the documentation for the DataPower WebGUI for information about the tasks that you can perform from the DataPower WebGUI.

Name

Specifies a user-defined symbolic name for a managed set.

A managed set name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # $ @ \ / , : ; " * ? < > | = + & % '.

Operational Status

Specifies the aggregated operational status of all the managed domains on all of the appliances in the managed set.

This field is read-only. The following values might appear in this field:

up Indicates that all of the service objects in all the managed domains on all of the appliances in the managed set are enabled.

partial Indicates that the service objects in all the managed domains on all of the appliances in the managed set are a mix of enabled, disabled and unknown.

down Indicates that all of the service objects in all the managed domains on all of the appliances in the managed set are disabled.

unknown
Indicates that the state of all of the all the managed domains on all of the appliances in the managed set cannot be determined. It is possible that there are no managed domains, or the state of the managed domains has yet to be retrieved, or that communication has been lost to the appliances.

Synchronization Status

Specifies whether the designated firmware version, sharable appliance settings, and managed domain versions for the managed set are synchronized across all of the appliances in the managed set. This synchronization status combines the firmware synchronization status, the settings synchronization status, and the synchronization status of all the managed domains on all the appliances of the managed set.

This field is read-only. The following values might appear in this field:

synced
Indicates that the managed set is synchronized.

changes pending
Indicates that the synchronization is queued. At least one appliance in the managed set has a firmware synchronization, settings synchronization, or managed domain synchronization queued.

in progress
Indicates that the synchronization is currently being synchronized. At least one appliance in the managed set has a firmware synchronization, settings synchronization, or managed domain synchronization in progress. Other firmware synchronizations, settings synchronizations, or managed domain synchronizations might also be pending for appliances in the managed set.

unknown
Indicates that the synchronization status is not known. At least one appliance in the managed set has a firmware synchronization, settings synchronization, or managed domain synchronization for which the synchronization state is unknown. Typically, this means that the appliance manager has not yet contacted the appliance to get the management status. Other firmware synchronizations, settings synchronizations, or managed domain synchronizations might also have changes pending, or in progress for appliances in the managed set.

error Indicates that an attempt to synchronize the managed set has failed because of an error on an appliance in the managed set. The appliance did respond, but responded with an error. Appliances in the managed set might also have a management status of unknown, in progress or changes pending for a firmware synchronization, settings synchronization, or managed domain synchronization. See the error log for the deployment manager for more information.

unreachable
Indicates that the appliance is unreachable. An attempt to synchronize the managed set has failed because an appliance is not responding. Appliances in the managed set might also have a management status of unknown, in progress, changes pending or error for a firmware synchronization, settings synchronization, or managed domain synchronization. See the error log for the deployment manager for more information.

Managed set settings

Use this page to view the general properties for the managed set, and a list of the appliances that are part of this managed set. You can also use this page to view lists of the shareable appliance settings versions, and managed and unmanaged domains that can be used with this managed set.

To view this administrative console page, click **Servers > DataPower > Managed sets >** *managedset_name*.

In addition to specifying values for a managed set, you can use the following buttons to edit appliance membership, synchronize appliances or access the DataPower WebGUI to configure an appliance.

Button	**Resulting action**
Edit Membership	Accesses a list of DataPower appliances that are members of the managed set. You can use this list to add appliances, remove appliances, or mark an appliance as the master appliance.
Synchronization	Forces a synchronization of the appliances in the managed set. The appliance manager attempts to maintain the synchronization automatically, but this option allows you to request that a synchronization be performed.

Button	Resulting action
Launch **gotcha:** You can also use the **Launch** button that is located in the Unmanaged domains section to launch the DataPower WebGUI in the default domain on the master appliance. The **Launch** button that is located in the Managed domains section provides a similar function. However, if you click that button, the DataPower WebGUI is launched in the managed domain on the master appliance instead of in the default domain.	Launches the DataPower WebGUI in the default domain on the master appliance in the managed set using the credentials that are defined in the DataPower appliance manager. Whenever you modify shareable appliance settings or a managed domain on the master appliance, the changes are distributed to every appliance in the managed set. Verify that pop-ups are enabled on your browser before you launch the DataPower WebGUI.

In the Firmware section, click **Change firmware** to change the firmware for the managed set. The firmware must have the same appliance type, model type and licensed features as the appliances in the managed set.

In the Settings section, click **Version history** to view the versions of shareable appliance settings that are available for this managed set

In the Appliance section, click **Appliance** to display a table of all of the appliances that are assigned to the managed set.

Name

Specifies a user-defined symbolic name for a managed set. This field is read-only

Firmware version

Specifies the firmware version that is to be synchronized to all of the appliances in the managed set.

If you have a single appliance managed set, and if there is not a firmware version in the DataPower appliance manager that matches the firmware that is used by the master appliance for that managed set, this field might be blank. In this situation, you can either add a firmware to the DataPower appliance manager that has the same version number, appliance type, model type and compatible licensed features as the single appliance in the managed set, or you can click **Change firmware** in the Firmware section, and select a firmware version for the managed set. This new firmware automatically becomes the firmware for the single appliance in this managed set, and must have the same appliance type, model type and licensed features as that appliance.

Operational status

Specifies the aggregated operational status of all the managed domains on all of the appliances in the managed set.

The status is unknown if the managed set does not have any managed domains. If the managed set has managed domains, the status reflects the aggregated operational status of all of the managed domains on all of the appliances in this managed set.

This field is read-only. The following values might appear in this field:

up Indicates that all of the service objects in all the managed domains on all of the appliances in the managed set are enabled.

partial Indicates that the service objects in all the managed domains on all of the appliances in the managed set are a mix of enabled, disabled and unknown.

down Indicates that all of the service objects in all the managed domains on all of the appliances in the managed set are disabled.

unknown
Indicates that the state of all of the managed domains on all of the appliances in the managed set

cannot be determined. This status might mean that the state has not yet been retrieved, or communication to the appliance has been lost. A managed set that does not have any managed domains always has a status of unknown.

Synchronization status

Specifies whether the designated firmware version, sharable appliance settings, and managed domain versions for the managed set are synchronized across all of the appliances in the managed set. This synchronization status combines the firmware synchronization status, the settings synchronization status, and the synchronization status of all the managed domains on all the appliances of the managed set.

This field is read-only. The following values might appear in this field:

synced
Indicates that the managed set is synchronized.

changes pending
Indicates that the synchronization is queued. The appliance has at least one firmware synchronization, settings synchronization, or managed domain synchronization queued.

in progress
Indicates that the synchronization is currently being processed. At least one appliance in the managed set has a firmware synchronization, settings synchronization, or managed domain synchronization in progress. Other firmware synchronizations, settings synchronizations, or managed domain synchronizations might also be pending for appliances in the managed set.

unknown
Indicates that the synchronization status is not known. At least one appliance in the managed set has a firmware synchronization, settings synchronization, or managed domain synchronization for which the synchronization state is unknown. Typically, this means that the appliance manager has not yet contacted the appliance to get the management status. Firmware synchronizations, settings synchronizations, or managed domain synchronizations might also be pending or in progress for appliances in the managed set.

error
Indicates that an attempt to synchronize the managed set has failed because of an error on an appliance in the managed set. The appliance did respond, but responded with an error. Appliances in the managed set might also have a management status of unknown, in progress or changes pending for a firmware synchronization, settings synchronization, or managed domain synchronization. See the error log for the deployment manager for more information.

unreachable
Indicates that the appliance is unreachable. An attempt to synchronize the managed set has failed because an appliance is not responding. Appliances in the managed set might also have a management status of unknown, in progress, changes pending or error for a firmware synchronization, settings synchronization, or managed domain synchronization. See the error log for the deployment manager for more information.

Managed domains

Specifies a list of managed domains. You can use this list to change a managed domain to an unmanaged domain.

The managed domains of a master appliance are copied to all appliances in the managed set. Expand this section to view a list of the managed domains. You can then select one or more managed domain, and click **Unmanage** if you want to turn the selected domains into unmanaged domains. When a managed domain becomes an unmanaged domain, it displays in the Unmanaged domains table instead of in the Managed domains table.

f you need to modify a managed domain on the master appliance, select that managed domain, and then click **Launch** to launch the DataPower WebGUI on the selected managed domain on the master appliance in the managed set. A new domain version is automatically created whenever you save changes in the managed domain on the master appliance. This new version becomes the current version and the changes are synchronized across the managed set.

Unmanaged domains on the master appliance

Specifies a list of unmanaged domains. You can use this list to manage these unmanaged domain.

The managed domains of a master appliance are copied to all appliances in the managed set. By default, domains are not managed. Unmanaged domains are not copied to subordinate appliances in the managed set.

Expand this section to view a list of the unmanaged domains. You can then select one or more unmanaged domain and click **Manage** to turn the selected domains into managed domains. When a unmanaged domain becomes a managed domain, it displays in the Managed domains table instead of in the Unmanaged domains table.

If you need to modify the configuration of the master DataPower appliance that is contains in an unmanaged domain, select that domain, and then click **Launch** to launch the DataPower WebGUI on the default domain of the master appliance in the managed set. You can then navigate to the unmanaged domain in the DataPower WebGUI. Clicking this **Launch** button is equivalent to clicking the **Launch** button at the top of this administrative console page.

Edit membership for a managed set

Use this page to add appliances to the managed set, remove appliances from the managed set, or mark an appliance as the master appliance for the managed set. All of the appliances in a managed set must have the same appliance type, model type and licensed features.

This page lists appliances that are members of this managed set.

To view this administrative console page, click **Servers > DataPower > Managed sets >** *managedset_name* **> Edit membership**.

To mark an appliance as the master appliance for the managed set, select the appliance that you want to make the master appliance, click **Mark as master**, and then click **Save**. After you click **Save**, the selected appliance becomes the master appliance.

To remove appliances from the managed set, select the appliances that you want to remove, click **Remove**, and then click **Save**. After you click **Save**, all the appliances that you selected are removed from the managed set.

Add appliances to the Managed Set.

Specifies a list of DataPower appliances that you can add to a managed set. Because all of the appliances in a managed set must have the same appliance type, model type and licensed features, only the appliances that meet these conditions are included in this list.

To add appliances to a managed set, select the appliances that you want to add from the list of appliances, and then click **Apply > Save**. After you click **Save,** all of the appliances that you selected are added to the managed set.

Appliance

Specifies the user-defined appliance name for a DataPower appliance. This field is read-only

Host name

Specifies an IP address or the fully qualified host name of a DataPower appliance. This field is read-only

Appliance type

Specifies the appliance type, such as XA35, XS40, and XI50. This field is read-only

Master

Specifies whether or not an appliance is the master appliance for the managed set. All appliances in the managed set are synchronized using the firmware version, settings, and managed domains of the master appliance. This field is read-only

Modifying DataPower appliance manager settings

You can change the global settings that apply to the DataPower appliance manager.

About this task

Complete the following actions to change the global settings that apply to the DataPower appliance manager.

Procedure

1. From the administrative console, click **Servers > DataPower > Appliance manager settings**.
2. To change the maximum number of domain and settings versions that can be stored in the DataPower appliance manager, specify a new value in the **Maximum number of versions to store** field.

   When the value specified in this field is exceeded, the DataPower appliance manager does not allow you to create any new versions until you delete one or more of the existing versions.

   gotcha: If you decrease the value for this setting, you might lose versions if the current number of versions stored exceeds the new maximum number of versions stored.
3. To change the directory where firmware versions, shareable appliance settings and managed domain versions are saved, specify thet new directory in the **Versions directory** field.

   gotcha: If you change the versions directory, you must move all of the current versions from the current directory to the new directory. This process might take a considerable amount of time, depending on how many versions you have, and could keep the appliance manager from performing other operations
4. View the status for the appliance manager.

   The DataPower appliance manager automatically starts when the deployment manager starts, if the appliance manager has appliances to manage. The appliance manager also automatically starts when a user submits a request to the appliance manager. For example, displaying the list of appliances that are defined in the appliance manager causes the appliance manager to start if it is not already running.
5. In the Operations section, click **Export** to export a copy of the DataPower appliance manager configuration to a file. This process is useful if you want to create backup copies of the appliance manager configuration.
6. In the Operations section, click **Import** to import the DataPower appliance manager configuration from a file. The file that you import from must reside on the same system as the deployment manager.
7. If you changed the maximum number of versions that you want stored, or you changed the version directory, click **Apply** to apply your changes.
8. Click **Shutdown Manager** if you need to stop the appliance manager.

Appliance manager settings

Use this page to specify global settings for the DataPower appliance manager. You can also use this page to shutdown the appliance manager.

To view this administrative console page, click **Servers > DataPower > Appliance manager settings**.

You can then click:

- **Export**, if you want to start the wizard that is used to export your DataPower appliance manager configuration to a file. After the wizard starts, you must specify the fully qualified name of the file into which you want to export the DataPower appliance manager configuration. The Export wizard exports the specified file onto the file system of the deployment manager.
- **Import** to start the wizard that is used to import a previously exported copy of your DataPower appliance manager configuration from a file. After the wizard starts, you must select either **Upload from local system** or **Upload from remote system**. Then specify the fully qualified file path of the file from which you want to import your DataPower appliance manager configuration. If you do not know the fully qualified name of the file, you can use the browse function to locate the file.
- **Shutdown Manager** if you want to shutdown the appliance manager.

Maximum versions stored

Specifies the maximum number of domain and settings versions that can be stored in the DataPower appliance manager. For example, if you specify a value of 3, for each managed set, you can have 3 sharable settings versions, and 3 versions of each managed domain.

You can view the shareable appliance settings version history, or the domain versions history to determine which versions are available for specific managed sets.

To view the sharable settings version history for a managed set, click **Servers** > **DataPower** > **Managed sets** > *managed_ set_name* > **Settings Version history**

To view the version history for a managed domain for a managed set, click **Servers** > **DataPower** > **Managed sets** > *managed_ set_name* > **Managed Domains** > .*managed_domain_name* > **Version history**

Versions directory

Specifies the directory into which you want the versions stored. Whenever the DataPower appliance manager detects a change in the sharable appliance settings, or a managed domain on the master appliance for a managed set, a new sharable appliance settings, or managed domain version is automatically created.

Status

Specifies whether the appliance manager is started or stopped. Click **Shutdown Manager** to stop the appliance manager.

DataPower appliance manager automatically starts when the deployment manager starts if there are appliances for the appliance manager to control. The appliance manager also automatically starts when requests are sent to the appliance manager.

If the status is stopped, the DataPower appliance manager automatically restarts the next time that you send a request to the appliance manager.

Importing an exported DataPower appliance manager configuration

You can import an exported DataPower appliance manager configuration if you want to use that configuration, instead of the current configuration, as the configuration for the appliance manager. For example, if the current configuration is corrupted, you might want to import a backup copy of that configuration that you have stored in either a local or remote file system.

About this task

Like a file that contains a firmware version, a file that contains an exported DataPower® appliance manager configuration can be stored on the system on which the browser is running, or it can be stored on the system on which the deployment manager is running. A file that is on stored on the system on which the browser is running is considered stored on a local system. A file that is stored on the system on which the deployment manager is running is considered stored on a remote system.

Complete these steps to use the administrative console to import a repository file from a remote system.

Procedure

1. In the administrative console, click **Servers > DataPower > Appliance manager settings**.
2. In the Operations section, click **Import** to import an exported DataPower appliance manager configuration.
3. Select **Upload from local system** or **Upload from remote system**.
4. In the **Fully qualified file name** field, enter the fully qualified file name for the file that you want to import. The fully qualified name of a file includes the full directory path to that file, and the file name. If you don't know the fully qualified name of the file, use the browse function to locate the file.
5. Click **Next**.
6. View the summary information, and then click **Finish**.

Exporting the DataPower appliance manager configuration

You can export the DataPower appliance manager configuration to a file on the system where the deployment manager is running.

About this task

Complete these steps if you want to use the administrative console to export the DataPower appliance manager configuration to a file on the system where the deployment manager is running.

Procedure

1. From the administrative console, click **Servers > DataPower > Appliance manager settings**.
2. In the Operations section, click **Export** to export the file to the system where the deployment manager is running.
3. In the **Fully qualified file name** field, enter the fully qualified file name for the file that you want to export. The fully qualified name of a file includes the full directory path to that file, and the file name.
4. Click **Next**.
5. View the summary information, and if the summary information is correct, click **Finish**.

Monitoring tasks that DataPower appliance manager is handling

Use this page to view the status of long running requests, or tasks, that are queued for the DataPower appliance manager to complete.

About this task

There are additional tasks that the appliance manager automatically runs that are used to maintain the state of the managed sets. These tasks are not included in the list of long running tasks. However, if an error occurs while one of these tasks are running, the error is recorded in the log to help you diagnose the problem.

Complete the following steps if you want to use the administrative console to view the long-running tasks that are in queue for the DataPower appliance manager.

Procedure

1. From the administrative console, click **Servers > DataPower > Tasks**.
2. From the Tasks collection page, view information about the tasks.

3. Optional: Select the Task ID of the task that you want to delete, and then click **Remove**.

   When you click **Remove**, the task is removed from your view, but the appliance manager still runs the task as soon as it is able to do so.

   Tasks are automatically deleted after 24 hours to save memory.

   Note: You cannot remove a task to stop the task from being completed. The only way to interrupt a running task, or prevent the appliance manager from running a task, is to shutdown the appliance manager. Shutting down the appliance manager terminates all running and queued tasks.

DataPower appliance manager tasks collection

Use this page to view the status of a task. A task is a long running request that you have asked the DataPower appliance manager to process. You can remove a task from the list of pending tasks. However, removing a task from that list does not prevent that task from running.

This page lists the long running tasks that need to be completed for an appliance. If you notice that a error has occurred during the processing of a task, you can find additional information about that error in the deployment manager log. This additional information might help you to diagnose the problem.

There are additional tasks that the appliance manager automatically runs that are used to maintain the state of the managed sets. These tasks are not included in the list of long running tasks. However, if an error occurs while one of these tasks are running, the error is also recorded in the deployment manager log.

To view this administrative console page, click **Servers > DataPower > Tasks**.

To remove a task, select the task identifier (ID) in the list of tasks, and click **Remove**.

Tasks are automatically deleted after 24 hours to save memory.

gotcha: Removing a task does not stop the task from being completed, nor does it delete the task. Even though you are no longer able to view the task, the appliance manager still executes the task when it gets to that task in the queue of tasks to execute. The only way to interrupt a running task, or prevent the appliance manager from running a task, is to shutdown the appliance manager. Shutting down the appliance manager terminates all running and queued tasks.

Task ID

Specifies an automatically generated ID number for a task.

Creation Date

Specifies the date on which the task was created

Description

Specifies information about the task.

Result

Specifies the result for the completed task, such as the serial number for an added appliance.

Created By

Specifies the username of the user who issued the request that resulted in the task being created. This field is blank if security is not enabled for the deployment manager.

Status

Specifies whether the task is completed, or is still in progress. Valid states include: Completed, Error, In progress, and Queued.

Administering managed domain versions

You can change the version of a managed domain that the DataPower appliance manager uses for a managed set, or you can copy a version of a managed domain to another managed set.

About this task

Each time domain settings are changed, DataPower appliance manager automatically creates a copy of the previous domain settings as a new domain version. You can use the administrative console to view a history of domain versions, change a different domain version to the active version, or copy a domain version to another managed set. Complete one or more of the following steps to administer managed domains.

Procedure

1. From the administrative console, click **Servers** > **DataPower** > **Managed sets** > *managedset_name*.
2. In the Managed domains section, expand Managed domains, and then click the name of the managed domain whose history you want to view.
3. Click **Versions** > **Version history** > **Details** to view a list of the versions of that managed domain.

   From the Domains history settings page, you can perform the following actions:

   a. Select a version, and click **Change Domain Version** to change the selected version to be the new active version
   b. Click a version number to view information about the version, or change the description for the version.
   c. In the Additional Properties section, click **Copy to another managed set** to copy the domain version to another managed set.

Domain history collection

Use this page to view the domain history for DataPower appliance manager. A domain version is an automatically generated copy of a managed domain.

To view this page, in the administrative console, click **Servers** > **DataPower** > **Managed sets** > *managed_set_name* > **Managed domains** > *domain_name* > **Versions** > **Version history** > **Details**.

The newest version of the domain is, by default, the active version. To change from the current domain version to a domain version in the list, select the version number in the list of available versions, and click **Change Domain Version**.

Version

Specifies a numeric value that represents a domain version. When you change the settings for a domain on the master appliance in a managed set, the appliance manager automatically detects the change and creates a copy of the change as a new domain version.

Date

Specifies the date when the domain version was created and stored in the DataPower appliance manager. Each time a domain is changed, a new version is created and stored in the DataPower appliance manager.

Description

Specifies information about the domain version. When a domain version is automatically generated, a default description is provided.

In Use

Specifies which version is active. The active version is automatically synchronized to all of the appliances in the managed set. Only one version can be active at a time.

Domain history detailed information

Use this page to view detailed information about a specific domain version for a managed domain in a managed set. A domain version is an automatically generated copy of an existing domain.

You can also use this page to specify a description for this domain version, or copy this domain version to a managed set.

To view this administrative console page, click **Servers > DataPower > Managed sets >** *managed_set_name* **> Managed domains >** *domain_name* **> Versions > Version history > Details >** *version_number*.

Version

Specifies the automatically generated version number of this domain version. This field is read-only.

Whenever you change a managed domain, the appliance manager automatically detects the change and creates a copy of the changed domain as a new domain version. Because the version number is automatically generated, you cannot change the value that appears in this field.

Date stored

Specifies the date when the domain version was created and stored in the DataPower appliance manager. This field is read-only.

Whenever a domain is changed, a new version is created and stored in the DataPower appliance manager. Because the date is automatically generated, you cannot change the value that appears in this field.

Description

Specifies information about the domain version. When a domain version is automatically generated, a default description is provided.

Managing versions of sharable appliance settings

Every time that sharable appliance settings are changed, the DataPower appliance manager automatically creates a copy of the new shareable appliance settings as a new settings version. You can view a list of all of the available settings versions that are available for an managed set, you can change which version is the active version, or you can copy a version to another managed set.

About this task

To manage the versions of sharable appliance settings for a managed set, in the administrative console, click **Servers > DataPower > Managed sets >** *managedset_name*, and then, in the Settings section, click **Version history**. Then complete one or more of the following actions if you want to display a list of all of the versions of sharable appliance settings that are available for a managed set, display information about a specific version, deploy a previous version, or associate a version with another managed set.

Procedure

- View the list of available versions.
- To change which sharable appliance settings version is the active version, select a version, and then click **Change Settings Version** . The DataPower appliance manager automatically synchronizes all of the appliances in the managed set to use the new active version.
- To view information about a specific version, click the name of that version.
- To change the description that is specified for a specific version, click the name of that version and update the **Description** field.

- To associate the selected version to another managed set, click the name of that version, and then, in the Additional Properties section, click **Copy to another managed set**.
  1. Using the wizard, choose a managed set from the list of managed sets, and then click **Next**
  2. View the summary information to verify that you have selected the correct managed set, and then click **Finish**

  The Copy to another Managed Set wizard starts and enables you to copy the selected version to a different managed set. The sharable appliance settings are applied to all appliances in this other managed set.

  gotcha: After the initial copy of the sharable appliance settings, the two managed sets are managed independently. Therefore, future changes to the sharable appliance settings in one managed set are not reflected in the other managed set.

  If you click **Servers > DataPower > Managed sets**, click the name of the managed set that you selected in the wizard, and then, in the Settings section, click **Version history**, you should see a new version as the active sharable appliance settings version for that managed set. The new active version is the version that the Copy to another Managed Set wizard created during the copy process.

What to do next

Changing the active settings version or copying the settings to another managed set causes a task to be created. If you want to monitor the progress of this task, in the administrative console, click **Servers > DataPower > Tasks**

Settings history collection for a DataPower appliance manager

Use this page to view the settings history for a DataPower appliance manager. A settings version is an automatically generated copy of the sharable appliance settings for the master appliance in a managed set.

To view this page, in the administrative console, click **Servers > DataPower appliances > Managed sets** > *managedset_name* **> Settings > Version history > Details**.

The newest version of the settings is, by default, the active version. To change from the current settings version to a settings version in the list, select the appropriate version number in the list, and click **Change Settings Version**.

Version

Specifies a numeric value that represents a settings version. When you change the shareable appliance settings for a master appliance in a managed set, the appliance manager automatically detects the change, and creates a copy of the change as a new settings version.

In Use

Specifies which version is active. The active version is automatically synchronized to all of the appliances in the managed set. Only one version can be active at a time.

Date stored

Specifies the date when the settings version was created and stored in the DataPower appliance manager.

Description

Specifies user-defined information about the settings version.

Settings version Collection

Use this page to view detailed information about a specific settings version for the appliances in a managed set. A settings version is an automatically generated copy of the shareable appliance settings for the appliances in a managed set.

You can also use this page to specify a description for this settings version, or you can click **Copy settings version to another managed set** to copy this settings version to another managed set. For example, if you have a managed set on a test system, and a different managed set on a production system, you can use this copy capability to ensure that you are using the same sharable appliance settings version on both managed sets.

When you copy a settings version to a different managed set, the copied settings version becomes the active version for the destination managed set.

To view this administrative console page, click **Servers > DataPower > Managed sets >** *managedset_name*, and then, in the Settings section, click **Version history**.

Version

Specifies a numeric value that represents a settings version. When you change the sharable appliance settings for the master appliance in a managed set, the appliance manager automatically detects the change, and creates a copy of the change as a new settings version. This is a read-only field.

Date stored

Specifies the date when the settings version was created and stored in the DataPower appliance manager. This is a read-only field.

Description

Specifies information about the settings version. When a settings version is automatically generated, a default description is provided.

Administering DataPower appliance domains

A DataPower appliance domain is a group of configuration information for an appliance. By default, these domains are unmanaged. You can use the administrative console to change an unmanaged domain to a managed domain, or to change a managed domain to an unmanaged domain. However, you cannot use the administrative console to configure a domain. You must use the DataPower WebGUI to configure a domain. See the DataPower WebGUI documentation for information about configuring a domain.

About this task

You can use the administrative console to change an unmanaged domain to a managed domain. However, you cannot use the administrative console to configure a domain. Use the DataPower WebGUI to configure a domain. See the DataPower WebGUI documentation for information about configuring a domain.

gotcha: The DataPower appliance manager synchronizes managed domains from the master appliance to the subordinate appliances in the managed set. However, it is possible that the managed domain might not be completely functional on all of the subordinate appliances. An example of this situation is when a service object, such as an XML firewall, in the managed domain has a listening port conflict on one of the subordinate appliances.

Complete one or more of the following steps to administer domains.

Procedure

1. From the administrative console, click **Servers > DataPower > Managed sets >** *managedset_name*.
2. To change an unmanaged domain to a managed domain, complete the following actions.
   a. In the Managed Domains section, expand Unmanaged domains.
   b. Select the unmanaged domains that you want to convert to managed domains, and then click **Manage**.
3. To change an managed domain to an unmanaged domain, complete the following actions.

   a. In the Managed Domains section, expand Managed domains.
   b. Select the managed domains that you want to convert to unmanaged domains, and then click **Unmanage**.
4. To access the DataPower WebGUI and re-configure a managed or unmanaged domain, select the domain that you want to re-configure, and click **Launch**.
5. To delete a managed domain, select the domain that you want to delete, and click **Delete**.

Results

If you changed any managed domains to unmanaged, those domains are now listed under Unmanaged domains. Similarly if you changed any unmanaged domains to managed, those domains are now listed under Managed domains.

Managed domain settings

Use this page to view the settings for a managed domain. You can also use this page to view the version history for this managed domain, or to view the managed domains that exist for a specific appliance.

DataPower appliances support the use of application domains. An application domain consists of resources that are configured to provide and support one or more services.

Domains are used to partition configuration information on the appliance into self contained units that make the information easier to manage. A managed domain is a domain on the master appliance that has been added to a managed set in the DataPower appliance manager. The DataPower appliance manager synchronizes the managed domain across all of the appliances in the managed set.

To view this administrative console page, click **Servers > DataPower > Managed sets >** *managedset_name* **> Managed domains >** *domain_name*.

Click **Launch** to access the DataPower WebGUI in the domain on the master appliance in the managed set. Pop-ups must be enabled to launch the DataPower WebGUI.

Click **Appliance** to display a table of all of the appliances that are assigned to the managed set and are synchronized for this managed domain. The table provides the following information for each appliance:
- The Name column specifies the name of the appliance in the appliance manager.
- The Master column indicates whether the appliance is the master in the managed set that the contains this managed domain.
- The Operational Status column indicates the operational status of the managed domain in the appliance. The following values might appear in this column:

  up Indicates that all of the service objects in the managed domain on the appliance are enabled.

  partial Indicates that the service objects in the managed domain on the appliance are a mix of enabled, disabled and unknown.

  down Indicates that all of the service objects in the managed domain on the appliance are disabled

  unknown
  Indicates the state of all of the in the managed domain on the appliance could not be determined. Possible causes are that the state has yet to be retrieved or communication has been lost to the appliance.
- The Synchronization Status column indicates whether the managed domain on the appliance is the same as the specified domain version for the managed set. The following values might appear in this column:

synced
Indicates that the managed domain on this appliance is the same as the specified domain version for the managed set.

changes pending
Indicates that the synchronization is queued for the managed domain on the appliance.

in progress
Indicates that the synchronization is currently being processed for the managed domain on the appliance.

unknown
Indicates that the synchronization status is not known. Typically, this means that the appliance manager has not yet contacted the appliance to get the synchronization status.

error
Indicates that an attempt to synchronize the managed domain on the appliance has failed because of an error on an appliance. The appliance did respond, but responded with an error. See the log for the deployment manager for more information.

unreachable
Indicates that the appliance is unreachable. An attempt to synchronize the managed domain has failed because the appliance is not responding. See the log for deployment manager for more information.

Click **Version History** to display a table of the versions for the managed domain. The table provides the following information for each version:

- The Version column specifies the version of the managed domain.
- The Date column indicates the date that the domain version was created.
- The In Use column indicates which domain version is active. The active domain version is the version this is synchronized across all of the appliances in the managed set.
- The Description column provides a description of the domain version.

Name

Specifies the name for the managed domain.

Secure Socket Layer communication with DataPower

Based on the default installations of the application server and the DataPower appliance manager, secure sockets layer (SSL) communication is used to send commands and receive events. The default SSL configuration used by the DataPower appliance manager can be strengthened by customizing the SSL connection. Modifying the default SSL configuration is optional and only needs to be done if the default configuration is not sufficient for your requirements.

SSL is used to send commands to each known appliance manager. In this scenario, the application server and the DataPower appliance manager behave as the SSL client and the DataPower appliances are acting as the SSL servers. This SSL connection uses the ibmPKIX trustmanager to do some verification of the DataPower appliance. Neither the certificate chain nor the revocation list for the certificate of the DataPower appliance are checked. The default configuration also does not do any SSL client validation for this scenario.

SSL is also used for the events received by the application server and the DataPower appliance manager from each DataPower appliance being managed. In this scenario, the application server and the DataPower appliance manager is the SSL server and the DataPower appliances are the SSL client. SSL client validation is also not performed in this scenario by default.

Most customizations that are available for SSL connections in WebSphere Application Server can be applied to the connections used by the DataPower appliance manager. To customize the SSL connections used for communication between the DataPower appliance manager and the DataPower appliances, each change made to the SSL connection on the DataPower appliance manager must also be accompanied by a complimentary change on each of the DataPower appliances that it manages. The DataPower appliance manager uses the **DataPowerMgr_sslConfig SSL** profile to connect with the DataPower appliances to send the appliances commands. You may make changes to this profile to influence the SSL connection that is used to send commands to the appliances. The DataPower appliance manager uses the **DataPowerMgr_inbound_secure** inbound endpoint on the Dmgr to receive events from the appliances it manages. You may make changes to the profile used by this endpoint to influence the SSL connection used to send events from the managed appliances.

Tip: For instructions on how to modify the AMP XML Management Interface SSL configuration on a DataPower appliance, refer to the DataPower appliance WebGUI Guide section on the XML Management Interface and how to create a custom SSL Proxy profile.

Adding the DataPower signer certificate to the WebSphere Application Server default truststore to enable an SSL connection

When configuring a DataPower appliance when security is enabled, the signer certificate of the DataPower server must be added to the WebSphere Application Server default truststore to enable an Secure Sockets Layer (SSL) connection to be made from WebSphere Application Server to the DataPower server.

About this task

You can add the signer certificate of the DataPower server to the WebSphere Application Server default truststore to enable an Secure Sockets Layer (SSL) connection using the administrative console or by using the addSignerCertificate wsadmin command.

The DataPower signer certificate should be installed in the `DataPower-root-ca-cert.pem` file under the Deployment managers profile in the `WAS_HOME/profiles/<DMGR profile>/etc` directory.

Procedure

1. From the administrative console, click **Security > SSL certificate and key management > Key stores and certificates > CellDefaultTrustStore > Signer certificates > Add signer certificate**.
2. In the Alias box, enter an alias name in which to identify the DataPower signer certificate.
3. In the File name box, enter the full path to the `DataPower-root-ca-cert.pem` file.
4. Click **Apply** and **Save**.

   Note: You can alternately use the addSignerCertificate wsadmin command to add the DataPower server to the WebSphere Application Server default truststore by entering the following:

   ```
   wsadmin> AdminTask.addSignerCertificate('[-keyStoreName
   CellDefaultTrustStore -certificateFilePath
   c:/wasHomeDir/profiles/Dmgr01/etc/DataPower-root-ca-cert.pem
   -certificateAlias datapower ]').
   ```

   If the `DataPower-root-ca-cert.pem` certificate file is not installed on the system, you can retrieve the DataPower certificate from the port using the administrative console:

   a. Click **Security > SSL certificate and key management > Key stores and certificates > CellDefaultTrustStore > Signer certificates > Retrieve from port**.
   b. In the Host box, enter the DataPower server hostname.
   c. In the Port box, enter the port of the DataPower server.
   d. In the Alias box, enter an alias name to identify the DataPower signer certificate.
   e. Click **Retrieve signer information**.
   f. Verify that the certificate information is correct, then click **Apply** and **Save**

Chapter 4. Setting up the proxy server

A *proxy server* is a specific type of application server that routes HTTP requests to content servers that perform the work. You can classify a proxy server according to the role that it plays in a system. This specific proxy server is classified as a *reverse proxy server* because the main function is to act as the first point of contact, not including the firewall, for client requests *into* the enterprise server. By contrast, a *forward proxy server* acts as the first point of contact for *outbound* traffic.

Before you begin

Review the topic that describes how to select a front end for your WebSphere Application Server topology. This topic helps you determine whether you should set up a web server plug-in, a proxy server, or a secure proxy server to provide session affinity, failover support, and workload balancing for your WebSphere Application Server topology.

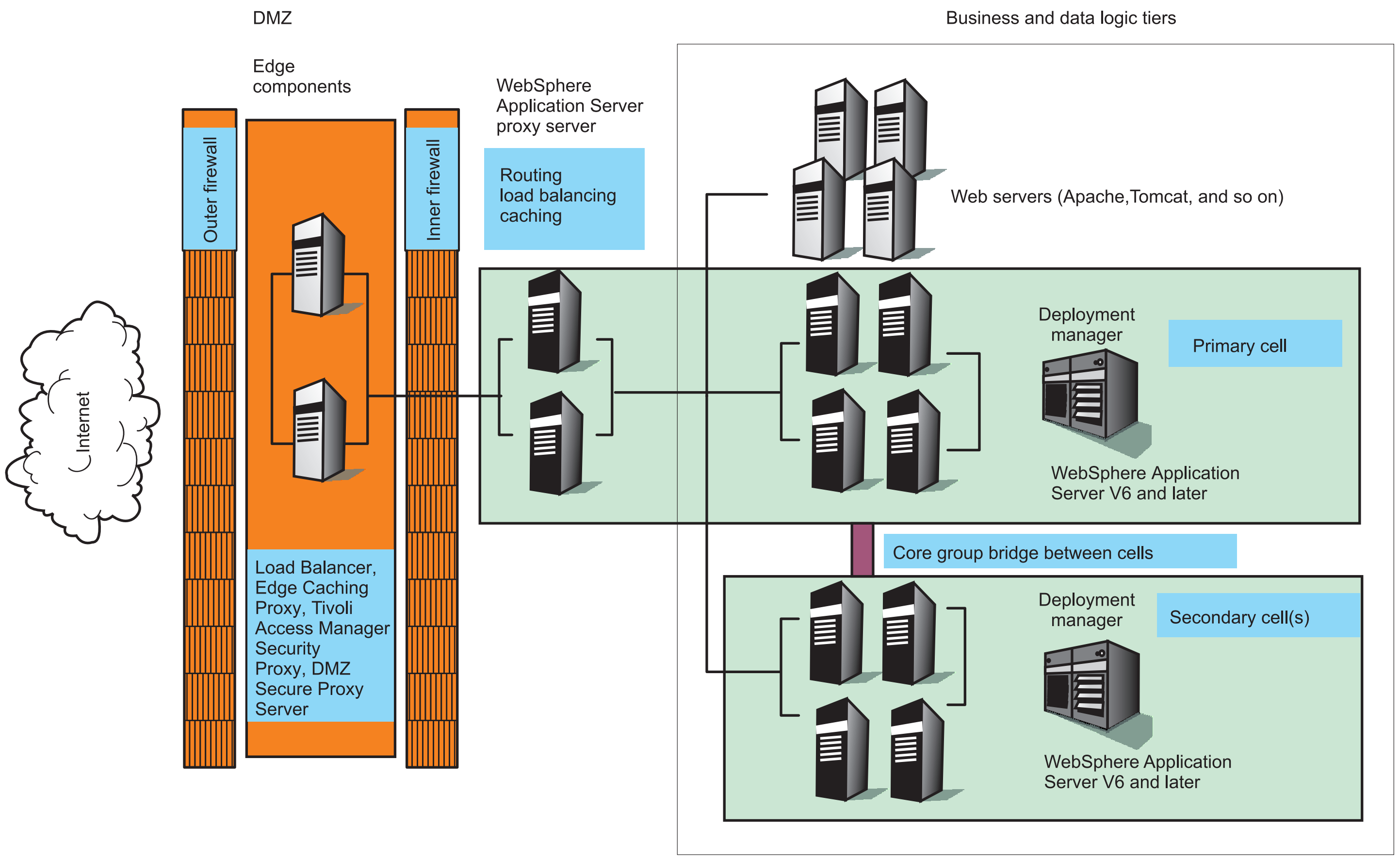
If you are familiar with the functionality of the legacy proxy server that existed in versions of the product prior to Version 6.0, review the following list of functions that were available in the legacy proxy server but are not available in the current proxy server:

- Forward proxy configuration, including SSL tunneling, transparent proxy, and FTP protocol
- Request URI rewrite (response URI rewrite is available)
- Authentication and authorization on the proxy server
- NCSA combined logging
- Custom logging
- Cache sharing
- Common Gateway Interface (CGI)

About this task

The proxy server acts as a surrogate for content servers within the enterprise. As a surrogate, you can configure the proxy server with rules to route to and load balance the clusters of content servers. The proxy server is also capable of securing the transport, using Secure Sockets Layer (SSL), and the content using various authentication and authorization schemes. Another important feature is its capability to protect the identity of the content servers from the web clients by using response transformations (URL rewriting). The proxy server can also improve performance by caching content locally and by protecting the content servers from surges in traffic.

A proxy server configuration provides settings that control how a proxy server can provide services for the enterprise applications and their components. This section describes how to create and configure proxy servers in an existing application server environment.


DMZ
Edge components
WebSphere Application Server proxy server
Business and data logic tiers
Outer firewall
Inner firewall
Routing load balancing caching
Web servers (Apache,Tomcat, and so on)
Internet
Deployment manager
Primary cell
WebSphere Application Server V6 and later
Load Balancer, Edge Caching Proxy, Tivoli Access Manager Security Proxy, DMZ Secure Proxy Server
Core group bridge between cells
Deployment manager
Secondary cell(s)
WebSphere Application Server V6 and later

In Version 6.0.2 of the product, you had to augment the deployment manager profile to manage the proxy server. For Version 6.1 and higher, the proxy server is managed from the administrative console without initial augmentation.

Procedure

- Create an HTTP or Session Initiation Protocol proxy server to route requests to application server nodes.
- Install a Session Initiation Protocol proxy server.
- Migrate profiles for the proxy server.
- Modify or add HTTP endpoints

What to do next

- Create proxy servers in the cell.
- Deploy applications to the proxy server.

Creating a proxy server

This topic provides information to create and configure a proxy server. When creating a proxy server, only the application server profile can be used as the target node.

About this task

The proxy server routes requests to application server nodes. The proxy server can dynamically route requests to all on-demand configuration (ODC) enabled application servers without additional configuration.

When you install the product, two profiles are created: a stand-alone application server profile, which is the default profile, and a deployment manager profile, which is called dmgr. When creating a proxy server, do not choose the deployment manager profile. Only the application server profile can be used as the target node.

Procedure

1. Create a proxy server in the administrative console by clicking **Servers > Server Types > WebSphere proxy servers > New**.
2. Select the node on which you want the proxy server to reside.
3. Enter a name for the new proxy server and click **Next**.
4. The supported protocols HTTP and SIP are selected for you. Select HTTP if your proxy server will route requests to and from a web container. Select Session Initiation Protocol (SIP) if your proxy server will route requests to and from a SIP container. Determine whether or not to generate unique HTTP ports by selecting or clearing Generate unique HTTP ports. Click **Next**.

   Note: If you create multiple proxy servers on the same node for vertical scaling, then you might select the option to generate unique ports to avoid port conflicts. Certain advanced scenarios pertain to port mapping that might require unique ports. For example, a load balancer can load balance requests to the proxy servers within the same node, assuming that each proxy server is listening on a unique HTTP port. For the proxy server to accept requests for a specific virtual host, it is necessary to add the unique HTTP ports that are generated to the host alias of the virtual host. It might also be necessary to modify the port values that the wizard generates, if these ports conflict with other local servers on the same node.
5. Select a proxy server template on which to base your proxy server. Click **Next**. You can select a default template, or you can choose to map to an existing proxy server.

   Mapping to a pre-existing proxy server is a time-saving technique. You can build one proxy server and apply all of the specific configurations your environment needs, and then use that proxy server as a template.
6. Review the summary panel and click **Finish**.

Results

You now have a functional proxy server that automatically routes HTTP requests to the cell that the proxy server belongs to or SIP requests to and from a SIP container. To enable routing to another cell, configure your cell to communicate with other cells.

gotcha: If the proxy server fails to start when attempting to start it as a non-privileged user on UNIX-based operating systems, change the ports of the PROXY_HTTP_ADDRESS and PROXY_HTTPS_ADDRESS transport chains to values greater than 1024.

gotcha: When using the proxy server, one should use the default port. If any other port is used, any HTTPS URL redirection might go to the wrong port. This is because the virtual host named default_host might also be using this new port. The HTTPS redirection is based on the first HTTPS port that it finds in the default_host.

gotcha: As the number of SIP containers grow in a deployment, the larger the heap settings need to be on the SIP proxy server. For example, a deployment of 20 containers requires a minimum heap size of 60 MB, so a -Xmo60m parameter should be added to the Generic JVM arguments field on the Java virtual machine panel of the admin console. However, a deployment of 70 containers requires a larger value such as 200 MB (-Xmo200m). See the information center topic *Java virtual machine settings* for more information about Generic JVM arguments.

Proxy server collection

This topic lists the proxy servers in the cell. A proxy server resides within a node.

A proxy server is used to classify, prioritize, and route HTTP and SIP requests to servers in the enterprise as well as cache content from servers. You can use this page to create, delete, or modify a proxy server.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers** .

To configure the proxy server to route work to WebSphere Application Servers in another cell, use core group bridge settings (**Servers > Core Groups > Core group bridge settings**), which sets up communication between cells.

Currently, configuring the proxy server to route work to a WebSphere Application Server Express cell requires advanced configuration. See the information on routing rules to learn more about setting the advanced configuration routing rules.

To configure the proxy server to route work to an application server that is not a WebSphere Application Server, the following advanced configuration is required:

1. Define a generic server cluster. From the administrative console, click **Servers > Clusters > Generic server clusters**.
2. Define a URI group. From the administrative console, click **Environment > URI groups**.
3. Create routing rules. From the administrative console, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name***> Proxy Server Properties > Routing rules**.

Both generic server clusters and URI groups are also accessible in the administrative console under Related Items for the proxy server.

Name

Specifies a logical name for the proxy server. For WebSphere Application Server, server names must be unique within a node.

If you have multiple nodes in a cluster, the server names must also be unique within the cluster. You cannot use the same server name within two nodes that are part of the same cluster. WebSphere Application Server uses the server name for administrative actions, such as referencing the server in scripting.

Node

The name of the node where the proxy server resides.

Version

Indicates the WebSphere Application Server version you are running.

Security level

The current overall security level of the proxy server.

The overall security level is determined based on the custom security settings. The possible values for Security level are High, Medium, Low, and Not applicable. The overall security level is equal to the security level of the setting that is considered the least secure. For example, to have an overall security level of High, all settings must be configured to the values associated with a HIGH level of security. If any of the settings are configured with a less secure value, the overall security level is the value of that setting.

Protocol

Indicates the protocol or protocols that the proxy server is configured to handle. This information is based on the types of transport channels that are included in the transport chains that are configured for the proxy server.

For example, if a transport chain includes an HTTP channel, `HTTP` displays in this field. If a transport chain includes both a SIP and an HTTP channel, `SIP,HTTP` displays in this field.

Status

Indicates whether the proxy server is started, stopped, or unavailable.

If the status is unavailable, the node agent is not running in that node and you must restart the node agent before you can start the server.

Table 21. Status settings of the proxy server.

This table lists the status settings of the proxy server.

➡	**Started**	The server is running.
✖	**Stopped**	The server is not running.
?	**Unknown**	Status cannot be determined. A server with an unknown status might, in fact, be running but has an unknown status because the application server that is running the administrative console cannot communicate with this server.

Proxy server configuration

You can modify an existing proxy server to perform advanced routing options, such as routing requests to a non-WebSphere Application Server cell, and to perform caching. The options to configure the proxy server from this panel are under Proxy server properties.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name*

Name

Indicates a logical name for the proxy server. Proxy server names must be unique within a node.

Run in development mode

Enabling this option can reduce the startup time of a proxy server. This time can include Java Virtual Machine (JVM) settings, such as disabling bytecode verification and reducing just-in-time (JIT) compilation costs. Do not enable this setting on production servers.

Specify this option if you want to use the -Xverify and -Xquickstart JVM settings on startup. After selecting this option, save the configuration and restart the proxy server to activate development mode.

The default setting for this option is `false`, which indicates that the proxy server is not started in development mode. Setting this option to `true` specifies that the proxy server is started in development mode with settings that speed server startup time.

Data type	Boolean
Default	false

Parallel start

Select this field to start the proxy server on multiple threads. This option might shorten the startup time.

Specify this option if you want the proxy server components, services, and applications to start in parallel, rather than sequentially.

The default setting for this option is `true`, which indicates that the proxy server is started using multiple threads. Setting this option to `false` specifies that the server does not start using multiple threads which might lengthen startup time.

The order in which the applications start depends on the weights that you assigned to each of them. Applications that have the same weight are started in parallel. You set the weight of an application with the **Starting weight** option on the **Applications > Application Types > WebSphere enterprise applications >** *application_name* page of the administrative console.

Data type	Boolean
Default	true

Start components as needed

Select this check box if you want the server components started as they are needed by an application that is running on this server.

When this check box is selected, server components are dynamically started as they are needed. When this check box is not selected, all of the server components are started during the server startup process. Therefore, selecting this option can improve startup time, and reduce the memory footprint of the server, because fewer components are started during the startup process.

Starting components as they are needed is most effective if all of the applications, that are deployed on the server, are of the same type. For example, using this option works better if all of your applications are web applications that use servlets, and JavaServer Pages (JSP). This option works less effectively if your applications use servlets, JavaServer Pages files and Enterprise JavaBeans™ (EJB).

Current Security Level

The current overall security level of the proxy server.

The overall security level is determined by evaluating the security settings. The possible values for Current Security Level are High, Medium, Low, and Not applicable. The overall security level is equal to the security level of the setting that is considered the least secure. For example, to have an overall security level of High, all settings must be configured to the values associated with a HIGH level of security. If any of the settings are configured with a less secure value, the overall security level is the value of that setting.

Proxy server settings

Use this topic to perform advanced configuration on a proxy server. Proxy settings enable the system administrator to fine tune the behavior of the proxy server. In particular, you can configure the connections and requests to the application server, enable caching, configure the requests that must be rejected, define how error responses are handled, and specify the location of the proxy logs.

The proxy server, upon creation, auto-senses the environment and is capable of routing requests to the product. Additional configuration can be applied to the proxy server to meet the needs of a particular environment.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> HTTP Proxy Server Settings > Proxy settings**.

You can edit configurable field settings for the proxy server on the Configuration tab.

Enable web services support

Specifies whether to enable the proxy server to route Web services traffic.

Data type	Boolean
Default	True

Static routing file directory

Specifies the directory on the proxy server where the static routing file is located.

Data type	String
Default	*profile_home*/`staticRoutes`

HTTP methods disabled

Specifies a list of HTTP methods that are disabled for the proxy server. Select the checkbox to enable this setting. Click **New** or **Delete** to add or remove HTTP methods from the list.

Data type	String
Default	Blank

Outbound connection settings

Specifies basic HTTP connection parameters between the proxy server and content servers.

`Outbound request timeout`
Specifies the default number of seconds the proxy server waits for a response before timing out a request to a content server. Consider this option carefully when changing the value.

`Outbound connect timeout`
Specifies the number of milliseconds that the proxy server waits to connect to a server. If this time expires, the proxy server attempts to connect to a different server. If no other available servers exist, the request times out. A value of 0 indicates that the proxy server should use the operating system kernel timeout value.

`Pool connections to content server`
Specifies the option to pool connections to the server is an optimization feature. Pooling prevents the need to frequently create and destroy socket connections to the server, by allowing the proxy server to pool these connections and reuse them.

`Maximum connections per server`
Specifies the maximum number of connections that will be pooled to any single content server.

`Local outbound TCP address`
Specifies the local outbound Transmission Control Protocol (TCP) address for data that enters and exits the SIP container. The value for this setting is the hostname or IP address to use for all communications between the SIP proxy and the SIP containers when the network is segmented.

Data type	String
Default	*
Range	IP address or valid host name

The following proxy custom properties are available to adjust the outbound connections.

- key=http.maxTargetReconnects: Maximum number of reconnects to the same target content server for each request. The default is 5.
- key=http.maxTargetRetries: Maximum number of times the proxy will attempt to select a new target content server for each request. The default is 5.
- key=http.routing.sendReverseProxyNameInHost: Determines whether or not the host header is rewritten for content that is not on a WebSphere Application Server content server. The options are `true` or `false` and are not case sensitive. If the value of this property is false, which is the default setting, then the host header is rewritten as the host of the target server. If the value for this property is true, then the host header is not rewritten.
- key=http.compliance.disable: Determines whether HTTP V1.1 compliance is enforced on proxy content server connections. The options are `true` or `false` and are not case sensitive. The default is false.

- key=http.compliance.via: The value of the via header that is appended to requests and responses for HTTP compliance. If the value is null, a via header will not be appended. If the value is true, a default via value is appended. Otherwise, the specified string via value is appended. The default is null.

Inbound connection SSL configuration

Specifies the SSL configuration from one of several sources.

Centrally managed
: When selected, specifies to use the SSL configuration that is scoped for this endpoint.

Specific to this endpoint
: When selected, enables the **Select SSL Configuration** list.

Select SSL Configuration
: Specifies a predefined SSL configuration.

Data type	String
Default	None
Range	NONE, CellDefaultSSLSettings, or NodeDefaultSSLSettings

Caching

Specifies whether to enable the proxy server to cache the content of servers.

When **Enable caching** is selected, static content caching is enabled for the proxy server, as defined by HTTP 1.1 specifications. By default, caching content is enabled.

The properties that follow apply only if caching is enabled:

Cache instance name
: Specifies the dynamic cache object cache instance that is configured in **Resources > Cache instances > Object cache instances**, which is used to cache all static and dynamic content responses. This object cache instance must be configured to support new I/O (NIO) application program interfaces (APIs).

Cache SSL content
: Determines whether client proxy server SSL connections that are terminated by the proxy server should have their responses cached.

Cache aggressively
: Enables caching of HTTP responses that would not normally be cached. Caching rules that are defined by HTTP 1.1 may be broken in order to gain caching optimizations.

Cache dynamic content
: Specifies whether dynamic content that is generated by WebSphere Application Servers V6.02 or later is cached. Caching dynamic content generated by content servers prior to WebSphere Application Server V6.02 is not supported.

Limit memory cache entry size
: When selected, the setting **Memory cache entry size** is enabled.

Memory cache entry size
: Specifies the maximum size of an individual cached response in MB. Any cached response larger than this will not be cached.

Logging

The proxy server has logs that are generated for proxy and stored cache requests. When **Enable access logging** is selected, you can specify the size and location of the access logs.

Access log maximum size
: Specify the maximum size, in megabytes, for an access log.

Data type	Integer
Units	Megabytes
Default	500

`Proxy access log`
Specifies a directory location for a proxy access log.

Data type	String
Default	${SERVER_LOG_ROOT}/proxy.log

`Cache access log`
Specifies a directory location for a cache access log.

Data type	String
Default	${SERVER_LOG_ROOT}/cache.log

`Local access log`
Specifies a directory location for a local access log.

Data type	String
Default	${SERVER_LOG_ROOT}/local.log

Note: There is a log called `${SERVER_LOG_ROOT}/local.log` that logs locally served proxy content. This content is not in the proxy cache.

HTTP requests are logged in one of three logs: proxy, cache, and local. Local log configuration is not currently available in the administrative console, but it is available at ${SERVER_LOG_ROOT}/local.log. Specify the location of this log by setting the http.log.localFileName custom property to the file location. The content of each log is formatted using National Center for Supercomputing Applications (NCSA) common log format.

- Proxy access log: Logs responses that are received from remote servers.
- Cache access log: Logs responses that are served from the local cache.
- Local access log: Logs all non-cache local responses, for example, redirects and internal errors.

Proxy custom properties that can be used to tweak logging are as follows:

- key=http.log.disableAll: This property disables all logging. A value of `true` stops proxy, cache, and local logging.
- key=http.log.maxSize:The maximum log size in megabytes (MB). A value of `UNLIMITED` indicates unlimited. 25 MB is the default.
- key=http.log.localFileName: Contains the name of the local log. A value of `NULL` indicates that the default `${SERVER_LOG_ROOT}/local.log` is used.

Security

Use this section to set up security options.

`Use a proxy-masking server header`
When selected, specifies to forward the content server's name to the client.

`Use the backend server header`
When selected, specifies the default server name is sent as the content server name.

`Specify a server header value`
When selected, the **Server header** setting is enabled.

Server header
Specifies the server name that is used in HTTP responses.

Trusted security proxies
Specifies intermediaries other than the proxy server to handle requests. This setting identifies which proxy servers can be trusted. WebSphere Application Server plug-in clients add private headers to the requests that they forward. For the proxy server to use those headers, the request must come from one of the trusted security proxies. If the request does not come from one of the trusted security proxies, then those private headers are ignored and removed from the request before the proxy server forwards the request. Use an IP or fully qualified host name in this field. If there are multiple IP addresses on the system where a WebSphere Application Server plug-in client is running, then the value in the trusted list must match the IP address of the outbound connection from that system. If you do not know the IP address that is used on the plug-in side of the connection, you should specify all of the IP addresses for that system to ensure that no matter which IP address is used on the outbound connection to the Proxy Server, that IP address matches one of the IP addresses in the trusted list.

Select the checkbox to enable **Security proxy**. Click **New** or **Delete** to add or remove proxies from the list.

Note: An empty list of trusted security proxies, which is the default value, indicates that no WebSphere Application Server plug-in clients are trusted.

Data type	String
Default	Blank
Range	IP address or valid host name

Proxy plug-in configuration policy

Use this section to configure proxy plug-ins.

Generate plug-in configuration
Specifies the generation of a proxy plug-in configuration file that you can use on a web server that is deployed in front of the proxy server. The plug-in can determine the URI that the proxy is handling on behalf of the application server. The plug-in can determine the endpoint, or boundaries of the proxy so that it can properly route requests that it receives to the proxy.

The options available to generate the plug-in are described in the following table:

Scope	Description
None	No scope.
All	The proxy server generates a plug-in configuration that includes all of the URIs that are handled by proxy servers in the local cell and all cells that are connected by a core group bridge.
Cell	The proxy server generates a plug-in configuration that includes all of the URIs that are handled by all the proxy servers in the cell.
Node	Includes all of the URIs that are configured for the node.
Server	The proxy server generates a plug-in configuration file only for the proxy server that is currently configured.

Plug-in config change script
Specifies the path to a script that is run after the WebSphere Application Server plug-in configuration is generated.

Custom error page policy

Use this section to configure settings for error pages when errors occur during the processing of a request.

The default is for no customized error pages to be generated.

`Error page generation application URI`
Specifies that if a valid uniform resource locator (URI) to an installed application is provided, the custom error page policy is enabled. If a valid URI to an installed application is not provided, the custom error page policy does not handle requests.

`Handle remote errors`
When selected, specifies HTTP response error status codes generated by the proxy server and HTTP response error status codes generated elsewhere after the proxy on the proxy content server connection error responses are handled. When not selected, only HTTP response error status codes generated by the proxy server are handled. A best practice is to configure an error page application on the same physical machine as the proxy server.

`Headers to forward to error page application`
Specifies additional header values from the client request to forward to the error page application as query parameters. The responseCode and URI query parameters are always sent to the error page application, in addition to the ones that are configured. The responseCode parameter is the HTTP status code that generates internally or is returned by the content server. The URI parameter is the request URI for the client.

Example - The error page URI is `/ErrorPageApp/ErrorPage`, the headers to forward contain `Host`, and a client sends the following request:

```
GET  /house/rooms/kitchen.jpg HTTP/1.1
Host:  homeserver.companyx.com
```

The request results in a HTTP 404 response (local or remote), and the request URI to the error page application would be:

```
/ErrorPageApp/ErrorPage?responseCode=404&uri=/house/rooms/kitchen.jpg&Host= homeserver.companyx.com
```

`HTTP status codes that are to be recognized as errors`
Specifies the status codes that the error page policy provides a response for. If a status code is not specified, the original content of responses with that status code are returned. If no HTTP status codes are specified, the defaults, `404` and `5XX`, are used. Instead of specifying status codes individually, the following method is recommended to represent a range:

- `5XX: 500-599`
- `4XX: 400-499`
- `3XX: 300-399`
- `2XX: 200-299`

Proxy custom property to use when tweaking the custom error page: `key=http.statuscode.errorPageRedirect`. This custom property determines whether error page generation is done using the redirect, instead of using the proxy error page application. The values are `true` or `false`. The default is `false`.

Static file serving

Specifies the values needed for the proxy server to perform static file serving.

`Static file document root`
Specifies the location on the file system where the static document files are located.

Data type	String
Default	${USER_INSTALL_ROOT}/staticContent

`Content mappings`

Specifies the content type mapping for a particular file extension. Specify a value for the following settings.

Extension	The subject file extension to map to a context type
Header	The header name to send to the client
Value	The value of the header to send to the client in the context-type header
Weight	A float value used to calculate the rank of files with this extension

Workload management

Specifies the values needed for the proxy server to perform workload management.

`High availability monitor timeout`

Specifies the amount of time, in seconds, before a high availability monitor timeout.

Data type	String
Units	Seconds
Default	300

`Advisor URI`

Specifies the uniform resource identifier (URI) for an advisor.

Data type	String
Default	/

`Load balancing algorithm`

Specifies the algorithm for the load balancer.

Data type	String
Default	Blank

Generic server clusters collection

Use this page to create, delete or modify a generic server cluster. Creating a generic server cluster is the next step towards generating the ability to route requests to a non-IBM application server, after creating the proxy server.

To view this administrative console page, click **Servers** > **Clusters** > **Generic server clusters**.

The system administrator can use the Generic Server Cluster panel to configure external servers, which are non-IBM application servers to create a logical cluster the proxy server can route work to. A generic server cluster defines the server endpoints that URI groups that are mapped to.

After you have created a generic server cluster, you want to create cluster members and define the cluster endpoints. Select the generic server cluster you created and click **Ports**.

Name

Specifies a logical name for the cluster. This is a user-defined field.

The name field cannot contain the following characters: `# \ / , : ; " * ? < > | = + & % '`

The name that defined must be unique among generic server clusters and cannot begin with a period or a space. A space does not generate an error, but leading and trailing spaces are automatically deleted.

Generic server clusters configuration

Use this topic to configure a generic server cluster. Creating a generic server cluster is the next step, after creating the proxy server, towards generating the ability to route requests to a non-IBM application server.

To view this administrative console page, click **Servers > Clusters > Generic server clusters >** *cluster_name*.

Now that you have configured a generic server cluster, you can create cluster members and define the cluster endpoints. Click **Additional properties** > **Ports**.

You can edit generic server cluster configurable field settings on the Configuration tab.

Name

Specifies the user-defined name for the cluster.

Protocol

The protocol field determines whether or not secure communication is used when connecting to members of the cluster.

The choices are HTTP and HTTPS.

Generic server cluster ports collection

After defining the generic server cluster name, use this page to create, delete, and configure the members of the cluster.

To view this administrative console page, click **Servers > Clusters > Generic server clusters >** *cluster_name***Ports**.

Host

The host name is either an IP address or the qualified host name of the cluster member. Specify a valid host name.

Port

Specify a valid port number to associate with the host.

Generic server cluster members

After defining the generic server cluster name, use this page to define the members of the cluster.

To view this administrative console page, click **Servers > Clusters > Generic server clusters >** *cluster_name* **> ports >** *host_name*.

After you complete this task, you can create a URI group and then define routing rules.

You can edit generic server cluster member field settings on the Configuration tab.

Host

The host name is either an IP address or the qualified host name of the cluster member.

Specify a valid host name.

Port

Enter the port on which the host name is listening. This entry ensures that the proxy server can communicate with the cluster member.

Specify a valid port number.

Weight

The load balancing weight is used to determine how frequently the cluster member is routed, relative to the other members in the group. The higher the weight the more frequently the cluster member is routed to.

The recommended weight value is between 1 and 20. A value of 0 will result in a cluster member that will never be routed to.

URI groups

A group of URI patterns, which you define, that can be mapped back to generic server clusters. When creating a URI group, verify that you are planning for URIs that form a logical collection. From this topic, you can create, delete, or modify a URI group.

To view this administrative console page, click **Environment** > **URI Groups**.

Name

The URI group name is a user-specified name.

The name field cannot contain the following characters: `# \ / , : ; " * ? < > | = + & % '`

The name that is defined must be unique among URI groups and cannot begin with a period or a space. A space does not generate an error, but leading and trailing spaces are automatically deleted.

URI group configuration

Use this topic to configure a URI group that can be mapped back to generic server clusters. When creating a URI group, ensure that you are planning for URIs that form a logical collection.

After you create a generic server cluster and a URI group, you are ready to define the routing rules that map the URI group to the generic server cluster. The routing rules ensure that the requests for specific URIs go to the proper generic server cluster.

To view this administrative console page, click **Environment** > **URI Groups** >*URI_group_name*.

You can edit URI groups configuration fields on the Configuration tab.

Name

The name field is required and is a user-defined field.

The name field cannot contain the following characters: `# \ / , : ; " * ? < > | = + & % '`

The name that is defined must be unique among URI groups and cannot begin with a period or a space. A space does not generate an error, but leading and trailing spaces are automatically deleted.

URI pattern

Use this field to define URI patterns that constitute the URI group.

The patterns can be any valid URI and can contain one or more wildcard characters.

Routing rules

Use this topic to set the advanced configuration routing rules to ensure work requests arrive at the proper generic server cluster. From this topic you can create, delete, or modify a routing rule.

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere proxy servers** > ***proxy_server_name*** > **> HTTP Proxy Server Settings > Routing Rules**.

Before you create routing rules, which are used to route requests to servers, you must define a generic server cluster (**Servers > Clusters > Generic server clusters**), a URI group (**Environment > > URI groups**), and optionally appropriate virtual hosts (**Environment > Virtual hosts > default host**).

Routing rules are used to assist the routing of work requests to non-IBM application server nodes. In addition, using routing rules, a system administrator can reroute work without heavily impacting the environment. This capability is useful when nodes are taken down for maintenance.

For example, the system administrator can set up a routing rule to route `/images/*` to the ImageServerCluster generic server cluster. If the **ImageServerCluster** cluster has to come down, the administrator can then route `/images/*` to another cluster with similar capability, or use a redirect rule. This situation explains why the URI group can be defined independently of the generic server cluster. If the generic server cluster must come down, the URI group can be rerouted elsewhere. When you create the generic server cluster by providing a name, you can configure the cluster by using the ports link to create the actual cluster members.

Routing rules function by using the configured virtual hosts and URIs as matching criteria. The proxy server scans all incoming requests and compares the URI and host header from it and matches it against the virtual host and URIs that are configured in the rule. You must create the URI group for a routing rule before creating the routing rule. If you are routing to a generic server cluster, you must also create the cluster before defining the routing rule. You can create the URI group by completing the following tasks:

1. Create the routing rule name.
2. Determine if you want to enable this rule. You can create routing rules and not enable them. This capability is useful when planning for the maintenance of nodes or for emergency planning.
3. Select the virtual host name from the drop-down menu. The virtual host name field is a selectable field that is preconfigured with the defined virtual hosts in the cell. If you do not see the virtual host that you want in the menu, click **Environment > > Virtual hosts** and define the host there.
4. Select the URI group for the routing rule. The URI group field is populated with all the preconfigured URI groups in the cell. If you do not see the URI group that you are looking for, click **Environment > > URI groups** and create one.
5. Select and define a routing rule. This option specifies how to route a request that matches the defined virtual host and URI group. The three options for this field are:
   - Generic Server Cluster: Routes requests to a preconfigured generic server cluster. Use the drop-down box to select the generic server cluster.
   - Fail: Rejects requests by returning the specified HTTP status code.
   - Redirect: Redirects a client to the specified URL. This option can be used to ensure a request is routed through Secure Sockets Layer (SSL).

Name

The name field is required and is a user-defined field.

The name field cannot contain the following characters: `# \ / , : ; " * ? < > | = + & % '`

The name that is defined must be unique among routing rules and cannot begin with a period or a space. A space does not generate an error, but leading and trailing spaces are automatically deleted.

Routing rules configuration

Use this topic to create the advanced configuration routing rules to ensure that work requests arrive at the proper generic server cluster.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> Proxy Server Properties > Routing Rules >** *rule_name*.

You can edit routing rules configurable field settings on the Configuration tab.

Name

The name field is required and is a user-defined field.

The name field cannot contain the following characters: `# \ / , : ; " * ? < > | = + & % '`

The name that is defined must be unique among routing rules and cannot begin with a period or a space. A space does not generate an error, but leading and trailing spaces are automatically deleted.

Enable this rule

You can create routing rules and then not enable them. This capability is useful when planning for the maintenance of nodes or emergencies.

By default, the rule is enabled.

Virtual host name

The virtual host name field is a selectable field that is preconfigured with the defined virtual hosts in the cell.

If you do not see the virtual host that you are looking for in the menu, click **Environment** > **Virtual Hosts** from the administrative console and define the virtual host there.

URI group

The URI group field is populated with all the preconfigured URI groups in the cell.

If you do not see the URI group that you are looking for, click **Environment** > **URI Groups** and create one.

Routing action

This option specifies to the proxy server how to route a request that matches the given criteria (virtual host and URI group) that you defined. You have three options for this field.

Generic Server Cluster: If you want only the proxy server to seek a preconfigured generic server cluster, select that option and use the drop-down box to select the generic server cluster.

Failure Status Code: If you want to reject the requests that match the specific criteria, you use the failure status code and provide an HTTP status code to use in the response to the sender.

Redirect URL: Use this option to send a redirect to the client. If you select this option, enter a fully qualified URL like `http://abc.xyz.com`. Usually, the URL is somewhere within the enterprise, sometimes right back to the proxy on a different port. You can use this option to ensure a request is routed through protocols like Secure Sockets Layer (SSL).

Local route: Use this option to route requests to the root directory on the file system where static files are located.

Rewriting rules collection

Use this page to define how to rewrite URLs in a request or response.

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere proxy servers** > *proxy_server_name* > **HTTP proxy server settings** > **Rewriting rules**.

From URL pattern

Specifies the original URL pattern in the 302 response header from the target server.

To URL pattern

Specifies how that URL should be modified so that the client is redirected to the proxy server or an appropriate public URL.

Rewriting rules configuration

Rewriting rules define how the proxy server rewrites URLs. Responses that have been redirected by target servers typically return a 302 status code with a location header that defines the URL that the client should be redirected to. Rewriting this URL is necessary if the target server is not aware of the proxy servers. The redirected URL is modified to correctly point clients to the proxy server instead of directly to a target server that may not be visible to clients. Use the following properties to configure the URL rewriting rules for a proxy server

gotcha: The proxy server only supports rewriting redirected responses. Therefore, these the following settings only apply to redirected responses. These settings do not apply to requests because the proxy server does not support URL rewriting for requests.

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere proxy servers** > *proxy_server_name* > **HTTP proxy server settings** > **Rewriting rules** > **New**.

From URL Pattern

Specifies the original URL pattern in the 302 response header from the target server.

The pattern can include the following wild card symbol: *

To URL Pattern

Specifies how that URL should be modified so that the client is redirected to the proxy server or an appropriate public URL.

The pattern can include the following wild card symbol: *

A rule with a from URL pattern of `http://internalserver/*` and to url pattern of `http://publicserver/*` would cause a redirected response with the original location header of `http://internalserver/secure/page.html` to be rewritten as `http://publicserver/secure/page.html`.

HTTP proxy inbound channel settings

Use this page to view and configure an HTTP proxy inbound channel. This type of transport channel provides the HTTP proxy capabilities.

To view this administrative console page, click **Servers** > **Proxy Servers** > *server_name* > **HTTP Proxy Server Settings** > **Proxy server transports** > **HTTP_PROXY_CHAIN** > **HTTP_proxy_inbound_channel_name**.

Transport channel name

Specifies the name of the HTTP proxy inbound channel.

The name field cannot contain the following characters: **# \ / , : ; " * ? < > | = + & % '**

This name must be unique across all channels in a WebSphere Application Server environment. For example, an HTTP proxy inbound channel and a TCP transport channel cannot have the same name if they reside within the same system.

Data type	String

Discrimination weight

Specifies the priority this channel has in relation to the other channels in this transport chain. This property is only used when port sharing is enabled and the channel chain includes multiple channels to which it might forward data. The channel in the chain with the lowest discrimination weight is the first one given the opportunity to look at incoming data and determine whether or not it owns that data.

Data type	Positive integer
Default	0

Starting a proxy server

Starting a proxy server starts a new server process based on the process definition settings of the current proxy server configuration.

Before you begin

Before you start a proxy server, verify that all of the resources that the proxy requires are available. You must also start all prerequisite subsystems.

About this task

This procedure for starting a server also applies to restarting a server. However, if a server fails and you want the recovery functions to complete their processing prior to new processes being started on the server, you must restart the server in recovery mode.

Use one of the following options to start a proxy server.

Procedure

- For the z/OS and distributed platforms, except AIX, you can issue the startServer command from the command line to start a single proxy server.

  You can issue the startServer command from the `C:\WebSphere\AppServer\profiles\AppSrv02\bin` directory.

  ```
  #  .\startServer.sh proxyserver1
  ```

  AIX You can issue the startServer command from the `/usr/WebSphere/AppServer/bin` directory.

  ```
  #  ./startServer.sh proxyserver1
  ```

- You can use the administrative console to start a proxy server.
  1. In the administrative console, click **Servers > Server Types > WebSphere proxy servers >** .
  2. Select the proxy server that you want to start, and click **Start**.
  3. Confirm that you want to start the proxy server.
  4. View the **Status** value and any messages or logs to see whether the proxy server was started.

Results

The specified proxy server starts. To verify that the proxy server was started, in the administrative console, click **Servers > Server Types > WebSphere proxy servers >** .

What to do next

If you need to start the proxy server with standard Java debugging enabled, then complete these steps:

1. From the administrative console, expand **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name*.
2. From Server Infrastructure, click **Java and process management > Process definition**.
3. Click **Java virtual machine**.

4. Select **Debug mode** to enable the standard Java debugger. Set **Debug mode** arguments, if needed, and click **OK**.
5. Save the changes to a configuration file.
6. Stop the proxy server, and restart the proxy server.

Stopping a proxy server

Stopping a proxy server ends a server process based on the process definition settings in the current application server configuration.

Before you begin

Ensure that you understand how stopping a particular server affects your ability to handle work requests, especially if you need to maintain a highly available environment.

About this task

There are times you need to stop a proxy server. For example, you might want to upgrade the operating system, or you might want to change a configuration setting for the proxy server. You can use one of the following options to stop a proxy server.

Note: To perform a proxy quiesce for your Session Initiation Protocol (SIP) proxy server, you must shut down the SIP proxy server by issuing the stopServer command from the command line. If you attempt to shut down the proxy server from the administrative console, then the server shuts down immediately and the proxy quiesce is not completed.

Procedure

- You can issue the stopServer command from the command line to stop a single proxy server.

  You can issue the stopServer command from the `C:\WebSphere\AppServer\profiles\AppSrv02\bin` directory.

  ```
  #  .\stopServer.sh proxyserver1
  ```

  AIX You can issue the stopServer command from the `/usr/WebSphere/AppServer/bin` directory.

  ```
  #  ./stopServer.sh proxyserver1
  ```

- You can use the administrative console to stop a proxy server.
  1. From the administrative console, click **Servers > Server Types > WebSphere proxy servers**.
  2. Select the proxy server, and click **Stop**.
  3. Confirm that you want to stop the selected proxy server.
  4. View the **Status** value and any messages or logs to see whether the proxy server stops.

Results

The specified proxy server stops as soon as requests assigned to that server finish processing. To verify that the proxy server is in the stop state, in the administrative console, click **Servers > Server Types > WebSphere proxy servers**.

Note: If the stopServer command is issued from the command line, then the server delays shutdown for a period of time until new inbound messages to route are not no longer being received. The quiesce feature notifies the load balancer to discontinue routing inbound messages by sending error responses to the advisor messages.

What to do next

If errors occur when you shut down a server, then read the *Troubleshooting and support* PDF.

By default, the SIP proxy server stops the flow of messages between the load balancer and the back-end containers to prevent calls from being lost when the proxy server shuts down. This process is called a proxy quiesce.

During proxy quiesce, the SIP proxy server notifies the load balancer and the back-end containers that the server is shutting down. After the devices stop forwarding messages through the proxy server, the server shuts down.

The default quiesce timeout period is three minutes. The SIP proxy server also waits a minimum of 20 seconds to allow the quiesce process to complete. The SIP proxy server continues to forward messages to the back-end containers while it responds to advisor messages from the load balancer with an error response. During a quiesce, the SIP proxy server also notifies the back-end containers that the proxy server is no longer a member of the cluster. After the initial 20 seconds, the SIP proxy server shuts down based on the specified amount of time configured for the proxy quiesce, which ranges from one second to a maximum of three minutes.

Complete these steps if you want to change the timeout period for proxy quiesce.

1. From the administrative console, expand **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name*.
2. From Server infrastructure, click **Java and Process management > Process definition**.
3. Click **Java virtual machine**.
4. Set the Generic JVM argument to `-Dcom.ibm.ejs.sm.server.quiesceTimeout=120`.
5. Define these SIP proxy custom properties to set the SIP proxy server to be fronted by a load balancer running a SIP advisor: **LBIPADDr** and **SIPAdvisorMethodName**.

HTTP proxy server custom properties

You can add the following custom properties to the configuration settings for an HTTP proxy server.

To specify custom properties for a specific HTTP proxy server, navigate to the custom properties page, and specify a value for the custom property.

1. In the administrative console, expand **Servers** > **Server Types** > **WebSphere proxy servers** > ***proxy_server_name*** to open the configuration tab for the server.
2. Expand **HTTP Proxy Server Settings**, click **Proxy settings**.
3. Under **Additional properties**, click **Custom properties** > **New**.
4. On the settings page, type the custom property to configure in the **Name** field, and type the value of the custom property in the **Value** field.
5. Click **Apply** or **OK**.
6. Click **Save** on the console task bar to save your configuration changes.
7. Restart the server.

The following list of HTTP proxy server custom properties is provided with the product. These properties are not shown on the settings page for an HTTP proxy server.

- "cache.ignore.header.Authorization" on page 243
- "cache.ignore.header.Cookie" on page 243
- "cache.ignore.header.Proxy-Authorization" on page 243
- "cache.query.string" on page 243
- "http.auto.redirect.correction" on page 243
- "http.cache.nocache.headers" on page 244
- "http.clientInfoFromTrustedIntermediary" on page 244
- "http.connectionPoolUseForPOST" on page 244

- “http.disable.retry.on.503.uriprefix” on page 244
- “http.disableresponsebufferingurls” on page 245
- “http.isDisable10ResponseCaching” on page 245
- “http.log.history” on page 245
- “http.maxCachedPayload” on page 245
- “http.odcUpdateTimeout” on page 246
- “http.pmiTimerInterval” on page 246
- “http.routing.sendReverseProxyNameInHost” on page 246
- “http.virtual.port.map” on page 246
- “HTTPProxyAdvisorMethodName” on page 247
- “HTTPProxyAdvisorStartupDelay” on page 247
- “HTTPProxyAdvisorURI” on page 247
- “HTTPProxyAdvisorUserAgent” on page 247
- “LBIPAddr” on page 248
- “localOutboundTCPAddress” on page 248

cache.ignore.header.Authorization

Specifies the proxy will ignore the Authorization header in the requests if you set the custom property to true. If the response is also able to be cached, proxy will cache the response.

Data type	Boolean
Default	false

cache.ignore.header.Cookie

Specifies the proxy to ignore the cookie header in the requests if you set the custom property to true. If the response is also able to be cached, proxy will cache the response.

Data type	Boolean
Default	false

cache.ignore.header.Proxy-Authorization

Specifies the proxy will ignore the Proxy-Authorization header in the requests if you set the custom property to true. If the response is also able to be cached, proxy will cache the response.

Data type	Boolean
Default	false

cache.query.string

Specifies whether the default proxy cache key generator uses both the URL path and query string information to generate the cache key. To enable this function, set the custom property to true. However, if you enable Edge Side Include (ESI) caching on the proxy server, this custom property is not used. If you do not set this custom property, the proxy server generates a cache key based on the URL path only and not based on the query string information.

For more information on ESI caching, see Configuring Edge Side Include caching.

Data type	String

http.auto.redirect.correction

Specifies whether the proxy server should try to correct the value in the location header when a 302 response is returned from the generic server cluster.

If you specify `none` for this property, the proxy server never tries to correct the value in the location header when a 302 response is returned.

If you specify `gsc` for this property, the proxy server tries to correct the value in the location header only when the 302 response is returned from the generic server cluster.

If you specify `managed` for this property, the proxy server tries to correct the value in the location header only when the 302 response is returned from a managed server.

If you specify `all` for this property, the proxy server tries to correct the value in the location header whenever a 302 response is returned.

gotcha: The junction rewrite rule overrides the value specified for this custom property.

Data type	String
Acceptable values	`none`, `gsc`, `managed`, or `all`
Default	`gsc`

http.cache.nocache.headers

Specifies the set-cookie headers that you do not want to cache when the proxy server receives a response. By default, the proxy server stores set-cookie headers in the proxy cache. If the Cache-Control header information is not set properly, the proxy server might store some session-related, user private cookies. You can use this custom property to specify which set-cookie headers you do not want to store in the proxy cache.

Data type	A comma-separated string of HTTP header names

For example, you might provide the following values: `Set-Cookie,Set-Cookie2`. As a result, the proxy server does not store Set-Cookie and Set-Cookie2 HTTP headers in the proxy cache.

http.clientInfoFromTrustedIntermediary

Specifies whether the proxy should extract the IP address from a WebSphere Application Server private header in a request. When the plug-in is deployed in front the proxy, the proxy server extracts the client IP address from the channels instead of from private headers forwarded from the plug-in. If this property is set to `true`, the proxy server extracts client information from private HTTP headers sent from the trusted plug-in instead of the channels.

Data type	String
Default	false

http.connectionPoolUseForPOST

Specifies whether the proxy server uses connection pooling for POST requests. POST requests, by default, are neither pooled, nor persistent requests. Therefore, if excessive POST requests are sent through the proxy server, port exhaustion might occur, resulting in bind exceptions. If this property is set to `true`, connection pooling is used for POST requests.

Data type	boolean
Default	false

http.disable.retry.on.503.uriprefix

Specifies URI prefixes for which you do not want the proxy server to automatically retry other servers when the proxy server receives a 503 response from the backend server to which it sent the initial request.

Typically when the proxy server receives a 503 response from a backend server, it marks the backend server as being down, and tries to send the request to another server. If, for specific URI prefixes, you do not want the proxy server to mark the backend server down, and to automatically retry other servers, you can specify those prefixes as the value of this custom property. For any URI prefix specified for this property, if the proxy server receives a 503 response to that request from the backend server, the proxy server returns a 503 response directly to the client instead of retrying the request. The proxy server also does not mark the backend server as being down.

Data type	string that consists URI prefixes separated by commas
Default	No default value

http.disableresponsebufferingurls

Specifies under what conditions a proxy server buffers a response for heartbeat applications.

The value of the property is a comma-separated URL pattern, such as `/`*`application_a`*`/.*html;/`*`application_b`*`/.*html`. For example, if the proxy server receives a `/`*`application_a`*`/heartbeat.html` request, which matches one of the URI patterns in the custom property, the proxy server does not buffer the response body for the request. In this example, if the proxy server returns a byte, it forwards the byte to the client side without buffering.

Without this custom property, the proxy server receives responses and buffers the response body until the partial response body limit is reached.

Important: The buffering process might improve proxy server performance, however, it might cause client-side connection time-outs for some heartbeat applications.

http.isDisable10ResponseCaching

Specifies whether the proxy should unchunk, and buffer the response for an HTTP 1.0 client.

If the http.isDisable10ResponseCaching property is set to `true`, the proxy server does not generate the content length header, and does not include chunked data in the client side response. Instead, the proxy server closes the connection at the end mark of the response body.

Data type	boolean
Default	false

http.log.history

Enables you to increase the number of history files for the proxy server log files.

The proxy server log files are the `proxy.log`, `local.log`, and `cache.log` files. Without this custom property, you have one history file for these proxy server log files.

Data type	Boolean
Default	1

http.maxCachedPayload

Specifies the maximum size of a chunked response, for which the proxy server will generate a content-length header.

When the proxy server receives a chunked response for an HTTP 1.0 client, the proxy server assumes that the HTTP 1.0 client cannot handle the chunked message, and tries to un-chunk the message, and calculate the content length for the response body. If the size of the chunked response is equal to, or less than the size limit specified for this property, the proxy server generates a content-length header for the response. If the chunked response exceeds the size limit specified for this property, the proxy returns a 500 error message.

Data type	integer
Default	100000 byes

http.odcUpdateTimeout

Specifies the amount of time, in seconds, for the HTTP proxy server to wait during server startup before routing information. The proxy server waits for the specified number of seconds before binding its ports.

This custom property can be used to configure a delay in startup (before the HTTP/HTTPS ports are bound) for a period of time to allow for the propagation of the routing information. If you set the value to `300`, the proxy server waits for 300 seconds to allow time for the routing information to propagate to the proxy server. If the routing information is propagated to the proxy server before 300 seconds, server startup resumes.

Data type	String
Default	150

http.pmiTimerInterval

Specifies the estimated time interval, in milliseconds, after which PMI statistics are recorded in the proxy server.

gotcha: Specifying a value that is lower than the default of 100 milliseconds might impact proxy server performance because the proxy needs to check the current time more frequently.

Data type	Integer
Default	100

http.routing.sendReverseProxyNameInHost

Specifies whether ‘the host header is rewritten for content that is not on a WebSphere Application Server content server.

The options for this property are `true` or `false` and are not case sensitive. If the value of this property is false, then the host header is rewritten as the host of the target server. If the value for this property is true, then the host header is not rewritten.

Data type	Boolean
Default	false

http.virtual.port.map

Specifies the server port mapping information that the backend server uses to send responses back to the correct external website ports.

When port mapping information is configured on a load balancer that is sitting in front of the proxy server, the proxy server might not be able to forward the correct external website port information to the backend server because the external website port number is probably different from the web server plug-in, or proxy server listening port. You can use this custom property to correct the port information before the proxy server passes it on to the backend server.

For example, if the proxy server, or the web server plugi-n is listening on ports 20080 and 20443, but the external website ports are 80, and 443, setting the following custom property for the proxy server enables the proxy server to provide the correct port information to the backend server. The backend server can then send responses to the correct external website ports.

```
Name:    http.virtual.port.map
Value:   20080:80;20443:443
```

Data type	String value of port mapping pairs separated by semicolons
Default	none

HTTPProxyAdvisorMethodName

Specifies the HTTP method name, which is commonly HEAD, GET for the methods used. The HTTP method name is also the advisor method set on the front end load balancer. The custom property is enabling the proxy to match incoming requests and determine whether the requests are from the load balancer or not.

Data type	String value (HTTP method name, such as GET or READ)
Default	none

HTTPProxyAdvisorStartupDelay

Specifies the HTTP Advisor start up delay in seconds. The default value is 0 seconds.

When the proxy receives a request, it tries to match the request against the custom properties (LBIPAddr, HTTPProxyAdvisorURI, HTTPProxyAdvisorURI, HTTPProxyAdvisorStartupDelay) if they are defined

Note: You do not need to define all of these custom properties.
If all of the custom properties are matched, the request is considered to be the advisor request from the front end load balancer.

If current proxy start time is less than the value defined in the HTTPProxyAdvisorStartupDelay custom property, an HTTP 503 error code is returned directly from the proxy server to inform the load balancer that the proxy/backend servers are not ready to service HTTP requests.

If current proxy start time is greater than the value defined in the HTTPProxyAdvisorStartupDelay custom property, the proxy forwards the advisor request to the backend and returns the response back to the front end load balancer. Based on the status code returned from the back-end server, the front end load balancer determines whether the proxy/backend server(s) are ready to service requests.

If the proxy is shutting down, the proxy server returns a 503 error code for the advisor requests. This error code informs the load balancer that the proxy and the backend servers are not able to service further requests and the front end load balancer needs to stop sending requests to this proxy server.

Data type	Integer (time in seconds)
Default	0

HTTPProxyAdvisorURI

Specifies a fully qualified URI string. This is also the advisor URI set on the front end load balancer. An application on the back-end server must be configured to answer this request URI. The custom property is enabling the proxy to match incoming requests and determine whether the requests are from the load balancer or not.

Data type	String value (fully qualified URI)
Default	none

HTTPProxyAdvisorUserAgent

Specifies the front end load balancer user agent name. This custom property is enabling the proxy to match incoming requests and determine whether the requests are from the load balancer or not.

Data type	String value (HTTP User-Agent header value)

Default	none

LBIPAddr

Specifies a semicolon separated IP address list. This custom property is allowing the proxy to match incoming requests and determine whether the requests are from the load balancer or not.

Data type	String value (IP addresses separated by semicolons)
Default	none

localOutboundTCPAddress

Specifies the host interface that is dedicated to HTTP traffic. This property determines the interface that is used to make outbound HTTP connections to the HTTP container.

Data type	String
Default	*

SIP proxy server custom properties

You can add the following custom properties to the configuration settings for the Session Initiation Protocol (SIP) proxy server.

To specify custom properties for a specific SIP proxy server, navigate to the custom properties page, and specify a value for the custom property.

Important: The custom properties are supported as the primary method of configuration. Therefore, if a custom property is set and then you set the corresponding setting in the administrative console, the custom property value is used.

1. In the administrative console, expand **Servers > Server Types > WebSphere application servers >** *server_name* to open the configuration tab for the server.
2. From **SIP proxy server settings**, expand **SIP proxy settings**, and click **Custom properties**.
3. From **Additional properties**, select **Custom properties** > **New**.
4. On the settings page, type the custom property to configure in the **Name** field, and type the value of the custom property in the **Value** field.
5. Click **Apply** or **OK**.
6. Click **Save** on the console task bar to save your configuration changes.
7. Restart the server.

You can define the following SIP proxy custom properties that are provided with the product. These properties are not shown on the settings page for a proxy server.

- “Overload custom properties” on page 249
- “burstResetFactor” on page 250
- “deflatorRatio” on page 250
- “dropOverloadPackets” on page 250
- “inDialogAveragingPeriod” on page 250
- “lsnLookupFailureResponseCode” on page 250
- “lsnLookupFailureReasonPhrase” on page 250
- “maxThroughputFactor” on page 250
- “numFailuresSipAdvisorRequests” on page 251
- “outDialogAveragingPeriod” on page 251
- “overloadResponseCode” on page 251

- “overloadResponseReasonPhrase” on page 251
- “perSecondBurstFactor” on page 251
- “proxyTransitionPeriod” on page 251
- “sipProxyStartupDelay” on page 252
- “sipAdvisorRequestTimeout” on page 252
- “trustedIPAddressList” on page 252

The following SIP general custom properties allow you to apply a variety of settings.
- “defaultTCPChainName” on page 252
- “defaultTLSChainName” on page 252
- “defaultUDPChainName” on page 253
- “identityAssertionHeaderRemovalEnabled” on page 253
- “isSipComplianceEnabled” on page 253
- “keepAliveFailures” on page 253
- “keepAliveInterval” on page 253
- “LBIPAddr” on page 254
- “localOutboundTCPAddress” on page 254
- “maxForwardsHeaderRequired” on page 254
- “maxWriteQueueEntries” on page 254
- “receiveBufferSizeChannel” on page 254
- “receiveBufferSizeSocket” on page 255
- “retryAfterValue” on page 255
- “sendBufferSizeSocket” on page 255
- “serverUDPInterface” on page 255
- “serverUDPPort” on page 255
- “SIPAdvisorMethodName” on page 256
- “sipClusterCellName” on page 256
- “tcp.IPSprayer.host” on page 256
- “tcp.IPSprayer.port” on page 256
- “tls.IPSprayer.host” on page 256
- “tls.IPSprayer.port” on page 256
- “udp.IPSprayer.host” on page 256
- “udp.IPSprayer.port” on page 256

Overload custom properties

Several of the SIP custom properties allow you to apply proxy-managed overload protection (PMOP). The PMOP overload settings allow for real-time protection against container overload.
- burstResetFactor
- deflatorRatio
- dropOverloadPackets
- inDialogAveragingPeriod
- maxThroughputFactor
- outDialogAveragingPeriod
- perSecondBurstFactor
- proxyTransitionPeriod
- sipProxyStartupDelay

- overloadResponseCode
- overloadResponseReasonPhrase

For more information on overload controls, refer to the Information Center topic *Session Initiation Protocol overload protection*.

burstResetFactor

Specifies the percentage of bursts during a given period of time. This custom property controls the amount of bursts that occur during the averaging period.

Data type	Integer
Default	100

deflatorRatio

Specifies a static ratio. This custom property is only used during the transition period when a transition period is specified.

Data type	Integer
Default	0

dropOverloadPackets

Specifies whether or not to drop packets when the SIP container is in an overloaded state. When this value is set to `False`, the proxy server responds with a 503 error when overloaded, otherwise the packet is dropped.

Data type	Boolean
Default	False

inDialogAveragingPeriod

Specifies the period of time, in seconds, during which in-dialog messages are averaged.

Data type	Integer
Default	180

lsnLookupFailureResponseCode

Specifies the response code when a SIP request for affinity fails. This custom property enables you to override the default 404 error response that normally displays when affinity fails.

Data type	Integer value that is a valid SIP response code
Default	404

lsnLookupFailureReasonPhrase

Specifies the response text to display when a SIP request for affinity fails. This custom property enables you to override the default "Not Found" error text that normally displays when affinity fails.

Data type	String
Default	"Not Found"

maxThroughputFactor

Specifies the percentage of the maximum number of messages per averaging period set. If this value is set to `0`, then the maximum throughput feature is disabled. This custom property is used to calculate the maximum number of messages allowed per second before the proxy server begins to reject requests for new sessions. This custom property should be set to the same value for each proxy server.

Data type	Integer
Default	0

numFailuresSipAdvisorRequests

Indicates the number of SIP advisor requests to miss before the SIP proxy recognizes that is it not processing any data. With this custom property, the SIP proxy will notice the failure to receive load balancer advisor requests.

If you use this custom property, consider setting its value to 3.

Data type	Integer
Default	No default value - this property is not enabled unless you specify a value.

outDialogAveragingPeriod

Specifies the period of time, in seconds, during which out-dialog messages are averaged.

Data type	Integer
Default	360

overloadResponseCode

Specifies the value for the response code returned from the proxy when overload occurs and SIP requests from the container are rejected. When the proxy is configured for overload protection, the SIP proxy can be configured to detect overload status. The proxy monitors the amount of traffic processed at the proxy and limits the number of new requests. If a container is overloaded, the proxy rejects requests with a 503 response code. If you prefer to use a different response code for overload protection, you can configure this custom property to return a different response code.

Data type	Integer
Default	503

overloadResponseReasonPhrase

Specifies the the response reason phrase that the proxy server issues when overload occurs and SIP requests from the container are rejected. When the proxy is configured for overload protection, the SIP proxy can be configured to detect overload status. The proxy monitors the amount of traffic processed at the proxy and limits the number of new requests. If a container is overloaded, the proxy rejects requests with a Service Unavailable response phrase. If you prefer to use a different response phrase, you can configure this custom property to return a different response phrase.

Data type	String
Default	Service Unavailable

perSecondBurstFactor

Specifies the percentage of bursts allowed through periodically (BTF).

Data type	Integer
Default	150

proxyTransitionPeriod

Specifies the period of time, in seconds, to lock the deflator after the container shuts down.

Data type	Integer

Default	0

sipAdvisorRequestTimeout

Specifies, in milliseconds, the amount of time the SIP proxy server waits for a DNS lookup to return from the Load Balancer.

If you do not specify a value for this custom property, the SIP proxy server waits for 2 seconds.

Specify a value of `0` to disable the monitoring of DNS lookups.

Data type	Integer
Default	3000 milliseconds

sipProxyStartupDelay

Specifies the period of time, in seconds, before the proxy server restarts to allow the proxy to become stable in the cluster and avoid an erroneous overloaded state.

Data type	Integer
Default	0

trustedIPAddressList

Specifies a colon-delimited list of IP addresses from which messages with a p-asserted identity header can flow through the SIP proxy server for WebSphere Application Server. If a p-asserted identity header exists in a message from an IP address that is not in this list of IP addresses, the header is removed.

The list of IP addresses must be specific, such as 192.168.0.1.

Beginning with this service release, the custom property requires a semicolon-delimited list. Also, you can specify a range of IP addresses such as 192.168.0.* instead of a list of specific IP addresses.

Data type	Numerical list of IP addresses
Default	None

defaultTCPChainName

Specifies the default TCP chain name to use when setOutboundInterface is not called. If you are using the web collaboration feature of the Feature Pack for Communications Enabled Applications (CEA) in a multihome environment, you must set this variable properly so that the web collaboration component can pick up the proper interface to use when routing data.

Data type	String
Default	None

defaultTLSChainName

Specifies the default TLS chain name to use when setOutboundInterface is not called. If you are using the web collaboration feature of the Feature Pack for Communications Enabled Applications (CEA) in a multihome environment, you must set this variable properly so that the web collaboration component can pick up the proper interface to use when routing data.

Data type	String
Default	True

defaultUDPChainName

Specifies the default UDP chain name to use when setOutboundInterface is not called. If you are using the web collaboration feature of the Feature Pack for Communications Enabled Applications (CEA) in a multihome environment, you must set this variable properly so that the web collaboration component can pick up the proper interface to use when routing data.

Data type	String
Default	None

identityAssertionHeaderRemovalEnabled

Specifies that all identity assertion related headers in SIP requests coming through the SIP Proxy to the SIP containers should be removed. If you set this property to `false`, the identity assertion related headers are kept as part of the requests.

Data type	Boolean
Default	true

isSipComplianceEnabled

Specifies whether SIP compliance checking is enabled in the SIP proxy server. SIP compliance checking ensures that the SIP messages conform to the Session Initiation Protocol standard. When this property is set to `true`, SIP compliance checking is enabled.

gotcha: If you are running a proxy server in a z/OS WebSphere Application Server, Network Deployment environment, and your proxy server is not part of a cluster, you can use this custom property to enable or disable SIP compliance checking for that SIP proxy server. However if you are running a stand-alone application server or your proxy server is part of a cluster, you must use the DisSipComplianceEnabledto generic JVM argument to enable or disable SIP compliance checking.

Data type	Boolean
Default	true

keepAliveFailures

Specifies the number of keepalive messages that must be missed before the proxy determines that the connection with the SIP container is down.

The proxy sends a keepalive message to the container at each keepAliveInterval. If the proxy does not receive a response to the message, it considers the lack of response as a failure. If the proxy receives a specific number of consecutive failures, it considers the container down and begins forwarding messages to a different SIP container.

Data type	Integer
Default	0

keepAliveInterval

Specifies the interval, in milliseconds, at which keepalive messages are sent to the SIP containers. A keepalive message is sent at the specified interval. If the specified number of **keepAliveFailures** messages is received from the SIP container, the proxy considers the container to be down. The proxy then routes data to a back-up SIP container until the connection between the proxy and the primary container is restored.

The first keepalive message contains the interval of time between the keep alive messages and the number of failures that are required before the container is considered down. The starting values should be specified based on the high availability (HA) heartbeat configuration.

Data type	Integer
Default	0

LBIPAddr

Specifies the IP address, such as `192.101.1.5`, of the load balancer used to load balance the SIP proxy. Multiple load balancer addresses can be configured by separating each IP address using a semicolon (;).

When SIP messages with the method configured as `SIPAdvisorMethodName` are received by the SIP proxy from the specified IP address, the SIP proxy responds with a success message if the SIP proxy can forward the messages to the SIP container. The SIP proxy responds with a failure message if messages cannot be forwarded to the SIP container. The load balancer then determines if the messages should be routed to the SIP proxy.

Data type	String
Default	null

localOutboundTCPAddress

Specifies the source interface to which the proxy binds when establishing connections to back-end SIP containers. This property is used when your proxy server is multihomed and needs to be configured to use a specific interface to send SIP traffic to the SIP containers. This property applies to both Transmission Control Protocol (TCP) and Transport Layer Security (TLS) connections.

Data type	String
Default	*

maxForwardsHeaderRequired

Specifies whether a Max-Forwards header must be present in all SIP requests and responses. The Max-Forwards header is used to limit the number of proxies or gateways that can forward a request.

The SIP Proxy requires that the Max-Forwards header be present in all SIP requests and responses. When this property is set to `true`, which is the default setting, and a Max-Forwards header is not included in a request, the SIP proxy issues a warning message that sends a 400 error response to that SIP request.

If you set this custom property to `false` the Max-Forwards header requirement is not enforced the requirement of the Max-Forwards header being required. Even if this property is set to `false` the SIP proxy decrements the value of this header if it is present in the request.

Data type	Boolean
Default	true

maxWriteQueueEntries

Specifies the number of messages that can be queued when a connection is slow or cannot be established. If the value is a large number, then more memory is consumed. A small number causes packets to be lost if the endpoint clears.

Data type	Integer
Default	100000

receiveBufferSizeChannel

Specifies the value, in bytes, for the maximum size of an incoming UDP packet, which is the size of the receive buffer that is allocated in the proxy server-side UDP connection.

Data type	Integer
Default	65535

receiveBufferSizeSocket

Specifies the value, in bytes, for the lower-level datagram buffers, which is the size of the DatagramSocket receive buffer (SO_RCVBUF) in the proxy server-side User Datagram Protocol (UDP) connection.

Use this property to buffer multiple packets in the DatagramSocket layer. If the value of the property is too small, then packets might be lost if the server is overloaded. If the value is too large, then the packets might be delayed.

Data type	Integer
Default	1024000

retryAfterValue

Specifies the amount of time, in seconds, before the client tries again to route a request to the proxy server. This custom property value is returned to the client in the error response if the SIP container is overloaded or if the SIP proxy cannot locate a server to which to route a request.

Data type	Integer
Default	5

sendBufferSizeSocket

Specifies the value, in bytes, for the lower-level datagram buffers, which is the size of the DatagramSocket send buffer (SO_SNDBUF) in the proxy server-side UDP connection.

Use this property to buffer multiple packets in the DatagramSocket layer. If the value of the property is too small, then packets might be lost if the server is overloaded. If the value is too large, then the packets might be delayed.

Data type	Integer
Default	1024000

serverUDPInterface

Specifies the host name or IP address that is used for all communications between the SIP proxy and the SIP containers when the network is segmented. This interface is the specific network interface for all UDP data that enters or exits the SIP containers. You must use this property with the serverUDPPort property.

Data type	String
Default	*

serverUDPPort

Specifies the UDP port that is used for SIP container communications. When the firewall is between the SIP proxy and the SIP container, you might set this value if a specific interface is needed to communicate with the SIP containers or if a specific port is required to pass through the firewall.

Data type	String
Default	dynamic

SIPAdvisorMethodName

Specifies a string value for the method sent by the load balancer to the SIP proxy for health checks.

The format is `OPTIONS` or `INFO`. This property is used with the LBIPAddr property.

Data type	String
Default	null

sipClusterCellName

Specifies the actual cell name that contains the cluster of SIP containers.

Set the **sipClusterCellName** custom property to be the cell name that contains the configured cluster of SIP containers

Data type	String
Default	Cell name in which the proxy resides

tcp.IPSprayer.host

Specified the host name for the IP Sprayer for Transmission Control Protocol (TCP) packets.

Data type	String
Default	None

tcp.IPSprayer.port

Specified the port for the IP Sprayer for Transmission Control Protocol (TCP) packets.

Data type	String
Default	None

tls.IPSprayer.host

Specified the host name for the IP Sprayer for Transport Layer Security (TLS) packets.

Data type	String
Default	None

tls.IPSprayer.port

Specified the port for the IP Sprayer for Transport Layer Security (TLS) packets.

Data type	String
Default	None

udp.IPSprayer.host

Specified the host name for the IP Sprayer for User Datagram Protocol (UDP) packets.

Data type	String
Default	None

udp.IPSprayer.port

Specified the port for the IP Sprayer for User Datagram Protocol (UDP) packets.

Data type	String
Default	None

SIP container custom properties

You can add any of the following custom properties to the configuration settings for a Session Initiation Protocol (SIP) container.

To specify custom properties for a specific SIP container, navigate to the custom properties page, and then specify a value for the custom property.

Important: The custom properties are supported as the primary method of configuration. Therefore, if a custom property is set and then you set the corresponding setting in the administrative console, the custom property value is used.

1. In the administrative console, expand **Servers > Server Types > WebSphere application servers >** *server_name* to open the configuration tab for the server.
2. From **Container settings**, expand **SIP Container settings**, and click **SIP container**.
3. From **Additional properties**, select **Custom Properties** > **New**.
4. On the settings page, type the custom property to configure in the **Name** field, and then type the value of the custom property in the **Value** field.
5. Click **Apply** or **OK**.
6. Click **Save** on the console task bar to save your configuration changes.
7. Restart the server.

The following list of SIP container custom properties is provided with the product. These properties are not shown on the settings page for the container.

You can define the following SIP container custom properties that are provided with the product. These properties are not shown on the settings page for the container.

- “auth.int.enable” on page 258
- “com.ibm.sip.sm.lnm.size” on page 258
- “com.ibm.webspehere.sip.security.digest.ldap.cachecleanperiod” on page 258
- “com.ibm.websphere.sip.security.tai.usercachecleanperiod” on page 259
- “com.ibm.ws.sip.key.set” on page 259
- “com.ibm.ws.sip.tai.DisableSIPBasicAuth” on page 259
- “DigestPasswordServerClass” on page 259
- “enable.system.headers.modify” on page 259
- “end.of.service.replication” on page 259
- “immediate.replication” on page 259
- “javax.servlet.sip.ar.dar.configuration” on page 260
- “javax.servlet.sip.ar.spi.SipApplicationRouterProvider” on page 260
- “javax.sip.bind.retries” on page 260
- “javax.sip.bind.retry.delay” on page 260
- “javax.sip.detect.pre.escaped.params” on page 260
- “javax.sip.force.connection.reuse” on page 261
- “javax.sip.hide.message.body” on page 261
- “javax.sip.hide.message.headers” on page 261
- “javax.sip.hide.request.uri” on page 261
- “javax.sip.OUTBOUND_PROXY” on page 262
- “javax.sip.PATH_MTU” on page 262
- “javax.sip.stat.report.interval” on page 262
- “javax.sip.trace.msg.in” on page 262

auth.int.enable

Specifies the **auth-int** quality of protection (QOP) for digest authentication. Digest authentication defines two types of QOP: **auth** and **auth-int**. By default, **auth** is used. When this custom property is set to True, the highest level of protection is used, which is the **auth-int** QOP.

Data type	String
Default	False

com.ibm.sip.sm.lnm.size

Specifies the number of logical names in the application server. Each SIP object that can be replicated, such as a SIP session, is associated with a logical name. All objects with the same logical name are replicated to the same back-up container. The proxy can route messages to the correct container using the logical name found in the message. The value must be greater than 1.

Data type	String
Default	10

com.ibm.webspehere.sip.security.digest.ldap.cachecleanperiod

Specifies the clean Lightweight Directory Access Protocol (LDAP) cache period in minutes.

Data type	String
Default	120

com.ibm.websphere.sip.security.tai.usercachecleanperiod

Specifies the clean security subject cache period in minutes.

Data type	String
Default	15

com.ibm.ws.sip.key.set

Specifies the key to use for SIP flow token security. When a value is specified for this property, SIP flow token security is automatically enabled.

Data type	String
Default	There is no default value

com.ibm.ws.sip.tai.DisableSIPBasicAuth

Specifies whether to allow basic authentication for SIP.

Data type	String
Default	False

DigestPasswordServerClass

Specifies types of user registries that are supported, except LDAP. To configure DigestTAI without the LDAP user registry, complete the following steps.

1. Create a class that implements this interface: com.ibm.ws.sip.security.digest.DigestPasswordServer
2. Add the following property to the SIP container custom property:

   Default LdapPasswordServer
3. Ensure that all users declared by the impl class are declared in the user registry configured for the product security.

Data type	String
Default	impl

enable.system.headers.modify

Specifies whether the application has access to headers that are otherwise restricted.

Data type	String
Default	False

end.of.service.replication

Specifies whether changes are buffered until the thread for a siplet is about to end. If the value is set to `true`, then each change is buffered until the thread for the siplet is about to end.

Data type	Boolean
Default	true

immediate.replication

Specifies whether each change is immediately sent to the Data Replication Service. When this property is set to `true`, when replication is issued from a non-SIP container thread, the replication is immediately performed on the calling thread. When this property is set to `false`, the changes are buffered, and replication does not occur until all changes are made.

Setting this property to true might have a negative impact on performance.

Data type	Boolean
Default	false

javax.servlet.sip.ar.dar.configuration

Specifies the location of the default application router (DAR) properties file. The properties file defines the order in which the application router sends SIP requests to applications as described in Appendix C of the JSR 289 specification.

Data type	String
Default	Null

javax.servlet.sip.ar.spi.SipApplicationRouterProvider

Specifies the custom application router implementation fully qualified class name as described in section 15.4.2 of the JSR 289 specification. The custom application router implementation class defines the order in which the application router sends SIP requests to applications.

Data type	String
Default	Null

javax.sip.bind.retries

Specifies the amount of time, in milliseconds, between attempts to start the SIP channel if the SIP port is busy with another process during server startup.

Data type	String
Default	60

javax.sip.bind.retry.delay

Specifies the delay, in milliseconds, between attempts to start the SIP channel if the SIP port is busy with another process during server startup.

Data type	String
Default	5000

javax.sip.detect.pre.escaped.params

Specifies whether to prevent the container from re-escaping Uniform Resource Identifier (URI) parameters that were pre-escaped by the application.

Enabling this property provides the application with more control over escaping URI parameters, when calling the **javax.servlet.sip.SipFactory.createURI()** and the **javax.servlet.sip.SipURI.setParameter()** parameters.

By default, the container only escapes characters that it must encode according to the RFC 3261 25.1 specification. In some cases, however, escaping additional characters might be required. Due to a limitation in the JSR 116 (5.2.1) specification, the application cannot perform its own escaping. Because of this limitation, attempts by the application to encode URI parameters causes the container to re-encode the percent sign. If the value of this property is set to true, the container cannot re-encode the percent sign.

Setting the value to true is not in compliance with the JSR 116 (5.2.1) specification, but provides the application with greater control over URI parameter escaping. APAR PK37192 describes the problem and the workaround.

Data type	String
Default	False

javax.sip.force.connection.reuse

Specifies whether to force reuse of inbound connections for outbound requests. This custom property is only relevant for stream transports, such as Transmission Control Protocol (TCP) and Transport Layer Security (TLS). Disabling this property causes the container to create a separate connection for outbound requests, even if an existing connection is already established to the same peer address. The connection is automatically reused if the top Via header in the inbound request contains an alias parameter. (http://www.ietf.org/internet-drafts/draft-ietf-sip-connect-reuse-07.txt)

Data type	String
Default	False

depfeat: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

mixv: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.hide.message.body

Specifies to hide message content in logs. Set the value of this property to `true` to remove the message body text from SIP messages printed in the log files. This property only affects the representation of the messages in log files.

Data type	String
Default	False

depfeat: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

mixv: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.hide.message.headers

Specifies to hide the specified message header field names in log files. The value of this property is a comma-separated list of header field names that you want removed from SIP messages printed in the log files. This property only affects the representation of the messages in log files.

Data type	String
Default	None

javax.sip.hide.request.uri

Specifies to hide request URIs in log files. Set the value of this property to `true` to remove request URIs from SIP messages printed in the log files. This property only affects the representation of the messages in log files.

Data type	Boolean
Default	False

javax.sip.OUTBOUND_PROXY

Specifies the fixed address for routing all outbound SIP messages. The format is `address:port/transport`, such as `1.2.3.4:5065/tcp`.

Note: Do not use this property if the container is fronted by an application server SIP proxy.

Data type	String
Default	null

depfeat: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

mixv: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.PATH_MTU

Specifies the maximum transmission unit, in bytes, for outbound User Datagram Protocol (UDP) requests. The SIP stack measures the size of a request before sending it out on the UDP channel. If the request is larger than the value specified for **PATH_MTU-200** (1300 bytes by default), then the transport is switched from UDP to TCP before transmission. Increase this value to send larger requests over the UDP channel; however, messages might be truncated or dropped. See the RFC 3261-18.1.1 specification for details.

Data type	String
Default	1500

depfeat: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

mixv: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.stat.report.interval

Specifies the amount of time, in milliseconds, for reporting dispatch and timer statistics to a system.out file. A value of zero indicates no report.

Data type	String
Default	0

javax.sip.trace.msg.in

Specifies whether to print incoming messages to a system.out file.

Data type	String
Default	False

javax.sip.trace.msg.out

Specifies whether to print outbound messages to a system.out file.

Data type	String
Default	False

javax.sip.transaction.invite.auto100

Specifies whether to automatically reply to invite requests with a `100 Trying` response. Disabling this property might increase the number of invite retransmissions.

Data type	String
Default	True

depfeat: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

mixv: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.a

Specifies, for UDP only, the amount of time, in milliseconds, before retransmitting invite requests for timer A for the RFC 3261 specification. This property is relevant for the invite client transaction.

Data type	String
Default	javax.sip.transaction.timer.t1

depfeat: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

mixv: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.b

Specifies the amount of time, in milliseconds, for the invite client transaction timeout timer (timer B) for the RFC 3261 specification.

Data type	String
Default	64*javax.sip.transaction.timer.t1

depfeat: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

mixv: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.cancel

Specifies the amount of timer, in milliseconds, for the timer to keep the cancelled client transaction in the proceeding state before completing the cancelled transaction for the RFC 3261 9.1 specification. This property is relevant for the invite client transaction.

Data type	String
Default	64*javax.sip.transaction.timer.t1

javax.sip.transaction.timer.d

Specifies the wait time, in milliseconds, before retransmission of the invite response for timer D for the RFC 3261 specification. This property is relevant for the invite client transaction.

Data type	String
Default	32000

depfeat: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

mixv: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.e

Specifies, for UDP only, the amount of time, in milliseconds, before the retransmission of the initial non-invite request for timer E for the RFC 3261 specification. This property is relevant for the non-invite client transaction.

Data type	String
Default	javax.sip.transaction.timer.t1

depfeat: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

mixv: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.f

Specifies the amount of time, in milliseconds, for the non-invite transaction timeout timer (timer F) for the RFC 3261 specification. This property is relevant for the non-invite client transaction.

Data type	String
Default	64*javax.sip.transaction.timer.t1

depfeat: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

mixv: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.g

Specifies the amount of time, in milliseconds, before retransmission of an initial invite response for timer G for the RFC 3261 specification. This property is relevant for the invite server transaction.

Data type	String
Default	javax.sip.transaction.timer.t1

depfeat: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

mixv: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.h

Specifies the amount of time, in milliseconds, to wait for an acknowledgement (ACK) receipt for timer H for the RFC 3261 specification. This property is relevant for the invite server transaction.

Data type	String
Default	64*javax.sip.transaction.timer.t1

depfeat: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

mixv: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.i

Specifies the amount of time in milliseconds to wait for an ACK retransmission for timer I for the RFC 3261 specification. This property is relevant for the invite server transaction.

Data type	String
Default	javax.sip.transaction.timer.t4

depfeat: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

mixv: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.invite.server

Specifies the amount of time, in milliseconds, for the timer to keep the invite server transaction in the complete state. This timer is not defined in the RFC specification.

To avoid creating a new server transaction when a client retransmits an invite request, keep the completed server transaction for a period of time before removing invite retransmissions. This timer is started when the transaction changes to the terminated state. When the timer completes, the transaction is removed.

Data type	String
Default	32000

javax.sip.transaction.timer.j

Specifies the amount of time in milliseconds to wait for non-invite request retransmission for timer J for the RFC 3261 specification. This property is relevant for the non-invite server transaction.

Data type	String
Default	64*javax.sip.transaction.timer.t1

depfeat: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

mixv: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.k

Specifies the amount of time, in milliseconds, to wait for non-INVITE response retransmissions for timer K for the RFC 3261 specification. This property is relevant for the non-invite client transaction.

Data type	String
Default	javax.sip.transaction.timer.t4

depfeat: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

mixv: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.non.invite.server

Specifies the amount of time, in milliseconds, for an Application Programming Interface (API) timer for the application to respond to a non-invite request. This property is relevant for non-invite server transactions.

This timer is not defined in the RFC specification. This property is needed to stop the transaction if the application does not generate a final response to the request. The timer starts when the request arrives in the stack and stops when a response is generated by the application. If no response is generated before the timer stops, then the transaction completes.

Data type	String
Default	34000

javax.sip.transaction.timer.t1

Specifies the amount of time, in milliseconds, for a network round trip delay for timer T1 for the RFC 3261 specification. The value is used as a base for calculating some timers and is relevant for all types of transactions, such as client, server, invite, and non-invite transactions.

Data type	String
Default	500

depfeat: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

mixv: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.t2

Specifies the maximum time in milliseconds before retransmitting non-invite requests and invite responses for timer T2 for the RFC 3261 specification.

Data type	String
Default	4000

depfeat: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

mixv: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.t4

Specifies the maximum amount of time, in milliseconds, for a message to remain in the network. This value is used as a base for calculating other timers for timer T4 for the RFC 3261 specification.

Data type	String
Default	5000

depfeat: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

mixv: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

on.outgoing.message.replication

Specifies whether changes are buffered until a siplet issues a request.send() or response.send() call. If the value is set to `true`, then each change is buffered until a siplet issues a request.send() or response.send() call.

Data type	Boolean
Default	false

pws_atr_name

Specifies the LDAP attribute name that stores the user password.

Data type	String
Default	userpassword

replicate.with.confirmed.dialog.only

Specifies whether to replicate the application session, even when no dialogs are confirmed. If the value is set to `false`, then the application session is replicated immediately after the session is created. Otherwise, the application session is only replicated when an associated dialog is confirmed.

Data type	String
Default	False

sip.container.heartbeat.enabled

Specifies whether or not SIP network outage detection is enabled for the SIP container. SIP network outage detection allows the SIP proxy to send keepalive messages to the SIP container if the value of this property is set to `true`.

If the value is set to `false` for the SIP container, then this property has no effect on the SIP proxy. However, if the value is set to `true` for the SIP container, the value should also be set to `true` for the SIP proxy to ensure that keepalive messages are answered at the SIP container and not presented to the application.

Data type	String
Default	true

sip.jsr289.parse.address

Specifies to use the SIP Servlet Specification 1.1, JSR 289 required format for createRequest() and createAddress() methods.

Note: The JSR 289 API requires that for any SIP URI that contains address parameters, you must enclose the SIP URI in angle brackets. The default behavior of the sip.jsr289.parse.address property is compliant with JSR 289 and correctly parses the address parameter as if it belongs to the SIP address. For example, when the property is set to `false`, the SIP address, `sip:fred@acme.com;param1=1`, is converted to `<sip:fred@acme.com;param1=1>`. When the property is set to `true`, the SIP address `sip:fred@acme.com;param1=1`, is converted to `<sip:fred@acme.com;>param1=1`.

Data type	String
Default	True

SIP_RFC3263_nameserver

Specifies whether to allow a SIP URI to be resolved through Domain Name System (DNS) into the IP address, port, and transport protocol of the next hop.

The value of the property is a string containing one or two address and port tuples, where two tuples are separated by a space. The following examples specify a one address and port tuple or a two address and port tuple.

dottedDecimalAddress@.port

hostname.domain@port

IPV6address@port

The following example values represent a single tuple.

- 1.2.3.4@53
- example.com@53
- a:b:c::d@53

The following example values represent two tuples separated by a space.

- 1.2.3.4@53 example.com@53
- a:b:c::d@53 9.32.211.14@53

Data type	String
Default	null

depfeat: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

mixv: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

thread.message.queue.max.size

Specifies the maximum number of events allowed in the container threads queue. When this number is exceeded, the proxy server is notified that the container is overloaded and requests for new sessions are not accepted. Instead, the container returns an error message that indicates that the container is temporarily unavailable.

This value represents the total number of messages for all queues and reflects the state of the CPU. When the CPU approaches 100%, the maximum value for this custom property is reached quickly. Configure your system to limit the queue size and prevent the queue from reaching this threshold.

Data type	String
Default	1000

weight.overload.watermark

Specifies the threshold value for the internal weight calculated by the container. When the container calculates the internal weight to be higher than the value specified, an overloaded container becomes available for service again.

This custom property represents a percentage of the maximum internal weight, such as 30 percent when the default value is set. When the high-water mark, or maximum threshold, is exceeded, the container waits until the weight drops below the maximum weight. This value cannot exceed `10`.

Data type	String
Default	3

Creating a proxy server cluster

A proxy server cluster consists of a group of proxy servers. You can use either the wsadmin tool, or the administrative console to create a cluster of proxy servers that can route requests to applications in a cell.

Before you begin

Member proxy servers in a proxy cluster can be created on any node in a cell and share the same proxy configuration. Before you define your proxy server cluster:

- Determine whether a domain name server (DNS), a load balancer, or a proxy server is going to be used to route requests to the proxy server cluster.
- Verify that Version 8.0 of the product is installed on the nodes that will belong to the proxy server cluster.

The main reason for creating a proxy cluster is to make it easier to administer multiple proxy servers. For example, if your proxy servers are members of a proxy server cluster, you can specify configuration settings for that proxy server cluster, and the settings are automatically applied to all of the proxy servers in that cluster. Creating a proxy server cluster also enables you to simultaneously start and stop all of the proxy servers in that cluster.

gotcha: A proxy server cluster does not have some of the same functionality that an application server cluster has. For example, because there is no data replication among members of a proxy cluster. failover between proxy servers in the cluster is not supported. If a proxy server is down, all active connections owned by the proxy server go away and then incoming requests fail. Both proxy servers and proxy clusters, on the other hand, support the high availability and failover of backend servers, such that the proxy server can detect if a backend server is down and then forward the requests to a server that has the session replicated.

About this task

Complete the following steps to create a proxy server cluster.

Procedure

Click **Servers > Clusters > Proxy server clusters > New** to start the Create a new proxy server cluster wizard.

Results

You have created a proxy server cluster.

What to do next

Click **Servers > Clusters > Proxy server clusters**, select the newly created cluster, and then click **Start** to start the cluster, or click on the name of the new cluster to change its configuration settings, or add additional members to the cluster.

Proxy server cluster collection

Use this page to view information about and change configuration settings for a proxy server cluster. A proxy cluster consists of a group of proxy servers.

To view this administrative console page, click **Servers > Clusters > Proxy server clusters** .

To define a new proxy server cluster, click **New** to start the Create a new proxy server cluster wizard.

Name

Specifies a logical name for the proxy cluster. The name must be unique among proxy clusters within the containing cell.

Supported Protocols

Indicates the protocol or protocols that the proxy server is configured to handle.

This information is based on the types of transport channels that are included in the transport chains that are configured for the proxy server. For example, if a transport chain includes an HTTP channel, HTTP displays in this field. If a transport chain includes both a SIP and an HTTP channel, SIP,HTTP displays in this field.

Status

This field indicates whether the proxy cluster is partially started, started, partially stopped, stopped, or unavailable. If the status is unavailable, the node agent is not running in that node and you must restart the node agent before you can start the proxy server.

After you click **Start** or **Ripplestart** to start a proxy cluster, each server that is a member of that proxy cluster launches if it is not already running. When the first member launches, the status changes to `partially started`. The status remains `partially started` until all proxy cluster members are running. When all proxy cluster members are running, the state changes to `running` and the status changes to `started`. Similarly, when you click **Stop** or **ImmediateStop** to stop a proxy cluster, the status changes to `partially stopped` when the first member stops, then changes to `stopped` when all proxy cluster members are not running.

Table 22. Status settings of the proxy server cluster.

This table lists the status settings of the proxy server cluster.

➡	**Started**	All of the proxy cluster members are running.
⇨	**Partially started**	At least one of the proxy cluster members is running.
✖	**Partially stopped**	At least one of the proxy cluster members has stopped running.
⊘	**Unavailable**	Status cannot be determined. All of the members of a proxy cluster with an unavailable status might, in fact, be running but the proxy cluster has an unavailable status because the proxy server that is running the administrative console cannot communicate with all of the proxy cluster members. The node agent should be started on all proxy nodes to ensure that the correct status is shown.

Proxy server cluster settings

Use this page to view a list of the proxy server clusters that are defined for your system, and the status of each of these clusters. You can also use this page to view or change the configuration settings for a specific proxy server cluster or to view the local topology of a specific proxy server cluster.

To view a list of the proxy server clusters that are defined for your system, and the status of each of these clusters, in the administrative console click **Servers > Clusters > Proxy server clusters**.

To display the topology of your proxy server clusters, click **Servers > Clusters > Proxy server clusters >** *proxy_server_cluster_name*, and then click the **Local Topology** tab.

To view or change the configuration settings of a proxy server cluster, in the administrative console click **Servers > Clusters > Proxy server clusters >** *proxy_server_cluster_name*.

Cluster name

Specifies a logical name for the proxy cluster. The name must be unique among proxy clusters within the containing cell.

Bounding node group name

Specifies the node group that forms the boundaries for this proxy cluster.

A node group is a collection of proxy server nodes. A node is a logical grouping of managed proxy servers, usually on a system that has a distinct IP host address. All proxy servers that are members of a proxy cluster must be on nodes that are members of the same node group. Nodes that are organized into a node group need enough capabilities in common to ensure that proxy clusters formed across the nodes in the node group can host the same application in each proxy cluster member. A node must be a member of at least one node group and can be a member of more than one node group.

Create and manage node groups by clicking **System administration > Node groups** in the administrative console.

Local Topology tab

Use this page to display, in a tree format, a list of all of the application server proxy clusters defined for your environment. The list shows all of the nodes and proxy cluster members that are included in each proxy cluster contained in a cell.

Proxy server cluster member collection

Use this page to view and manage proxy servers that belong to a proxy cluster.

To view this administrative console page, click **Servers > Clusters > Proxy server clusters >** *proxy_cluster_name* **> Cluster members**.

Proxy servers that are part of a proxy cluster are referred to as proxy cluster members. A copy of the first proxy cluster member that you create is stored as part of the proxy cluster data and becomes the template for all additional proxy cluster members that you create.

You can use this page to perform the following actions:

- Create a New proxy cluster member. To create a new member, click **New**.

  A copy of the first proxy cluster member that you create is stored as part of the proxy cluster data and becomes the template for all additional proxy cluster members that you create. Any individual configuration change that you make to a proxy cluster member does not affect the configuration settings of the proxy cluster member template.

You can use wsadmin commands to modify the proxy cluster member template, or you can click **Servers** > **Clusters** > **Proxy server clusters** > *proxy_cluster_name* > **Cluster members**, and then click **Templates**. Any change that you make to the template does not affect existing proxy cluster members.

- Delete a proxy cluster member. To delete a proxy cluster member, select the proxy cluster member you want to delete, then click **Delete**.
- Create or edit proxy cluster templates. To work with proxy cluster templates, click **Templates**.
- Start a proxy cluster member. To start a proxy cluster member, select the server that you want to start, then click **Start**.
- Stop a proxy cluster member. To stop a proxy cluster member, select the server that you want to stop, then click **Stop**.

Member name

Specifies the name of the server in the proxy cluster. On most platforms, the name of the server is the process name. The name must match the (object) name of the proxy server.

Node

Specifies the name of the node for the proxy cluster member.

Version

Specifies the version of the product on which the proxy cluster member runs.

Status

This field indicates whether the proxy cluster member is started, stopped, partially stopped, or unavailable. If the status is unavailable, the node agent is not running in that node and you must restart the node agent before you can start the server.

Table 23. Status settings of the proxy server cluster member.

This table lists the status settings of the proxy server cluster member.

➔	**Started**	The proxy cluster member is running.
⊗	**Partially stopped**	The proxy cluster member is in the process of changing from a started state to a stopped state.
✖	**Stopped**	The proxy cluster member is not running.
⊘	**Unavailable**	Status cannot be determined. A proxy cluster member with an unavailable status might, in fact, be running but has an unavailable status because the proxy cluster member that is running the administrative console cannot communicate with this proxy cluster member.

Proxy cluster member settings

Use this page to manage the members of a proxy cluster. A proxy cluster of proxy servers is managed together and participate in workload management.

To view this administrative console page, click **Servers** > **Clusters** > **Proxy server clusters** > *proxy_cluster_name* > **Cluster members** > *cluster_member_name*.

Member name

Specifies the name of the proxy server in the proxy cluster. On most platforms, the name of the server is the process name. The member name must match the name of one of the servers that are listed on the proxy servers page.

Run in development mode

Enabling this option may reduce the startup time of an proxy server. This might include Java virtual machine (JVM) settings, such as disabling bytecode verification and reducing Just in Time (JIT) compiler compilation costs. Do not enable this setting on production servers.

Specifies that you want to use the JVM settings, **-Xverify** and **-Xquickstart**, on startup. After selecting this option, save the configuration and restart the server to activate development mode.

The default setting for this option is `false`, which indicates that the server is not started in development mode. Setting this option to `true` specifies that the server is started in development mode, using settings that decrease server startup time.

Data type	Boolean
Default	false

Parallel start

Specifies whether to start the server on multiple threads. When you start the server on multiple threads, the server components, services, and applications start in parallel rather than sequentially, which might shorten the startup time.

The default setting for this option is `true`, which indicates that the server uses multiple threads when it starts. Setting this option to `false` specifies that the server uses a single thread when it starts, which might lengthen startup time.

Data type	Boolean
Default	true

Start components as needed

Select this property if you want the proxy server components started as they are needed by an application that is running on this proxy server.

When this property is selected, proxy server components are dynamically started as they are needed. When this property is not selected, all of the proxy server components are started during the startup process. Therefore, selecting this property usually results in improved startup time because fewer components are started during the startup process.

gotcha: If you are running other WebSphere products on top of the Application Server, make sure that those other products support this functionality before you select this property.

Proxy cluster member templates collection

Use this page to view the list of proxy cluster member templates that exist for this proxy cluster.

To view the proxy cluster member templates that are available for creating a new member of a proxy cluster, in the administrative console, click **Servers > Clusters > Proxy server clusters >** *proxy_cluster_name* **> Cluster members > Templates**.

To edit the attributes of an existing proxy cluster member template, click the name of the template.

Usually, only one template exists that you can use to create additional members for a proxy cluster. However, if a proxy cluster includes nodes that are at different versions of the product, a different template exists for each of these versions. For example, if a proxy cluster has proxy cluster members residing on both a Version 6.1 node and a Version 7.0 node, the proxy cluster has two templates. The Version 6.1

template is used when you create an additional proxy cluster member on the Version 6.1 node, and the Version 7.0 template is used when you create an additional proxy cluster member on the Version 7.0 node.

If you modify a template, all new proxy cluster members are created with the server property settings of the modified template. However, the property settings of existing proxy cluster members do not change. If you make any change to a proxy cluster member template, you should make the same change to all of the existing proxy cluster members.

Name
Specifies the name of the proxy cluster member template.

Platform
Specifies the operating system platform to which this template applies.

Version
Specifies the version of the product to which the template applies.

Description
Specifies a description of this proxy cluster member template. This field is optional and might be blank.

Proxy cluster member template settings
Use this page to modify proxy cluster member template attributes.

To view this administrative console page, click **Servers > Clusters > Proxy server clusters >** *proxy_cluster_name* **> Cluster members > Templates >** *cluster_member_template_name*. The following attributes are displayed:

Name
Displays the name of the proxy cluster member template.

Run in development mode
Enabling this option may reduce the startup time of a proxy server. This might include Java virtual machine (JVM) settings, such as disabling bytecode verification and reducing Just in Time (JIT) compiler compilation costs. Do not enable this setting on production servers.

Specifies that you want to use the JVM settings, **-Xverify** and **-Xquickstart**, on startup. After selecting this option, save the configuration and restart the server to activate development mode.

The default setting for this option is `false`, which indicates that the server is not started in development mode. Setting this option to `true` specifies that the server is started in development mode, using settings that decrease server startup time.

Data type	Boolean
Default	false

Parallel start
Specifies whether to start the proxy server on multiple threads. When you start the proxy server on multiple threads, the proxy server components, services, and applications start in parallel rather than sequentially, which might shorten the startup time.

The default setting for this option is `true`, which indicates that the proxy server uses multiple threads when it starts. Setting this option to `false` specifies that the proxy server uses a single thread when it starts, which might lengthen startup time.

Data type	Boolean
Default	true

Start components as needed

Select this property if you want the proxy server components started as they are needed by an application that is running on this proxy server.

Note: When this property is selected, proxy server components are dynamically started as they are needed. When this property is not selected, all of the proxy server components are started during the startup process. Therefore, selecting this property usually results in improved startup time because fewer components are started during the startup process.

Avoid trouble: If you are running other WebSphere products on top of the Application Server, make sure that those other products support this functionality before you select this property.

Creating a proxy cluster: Basic proxy cluster settings

Use this page to enter the basic settings for a proxy cluster.

To view this administrative console page, click **Servers > Clusters > Proxy server clusters > New**.

Proxy cluster name

Specifies the name of the proxy cluster. The proxy cluster name must be unique within the cell.

Supported Protocols

Indicates the protocol or protocols that the proxy server is configured to handle.

This information is based on the types of transport channels that are included in the transport chains that are configured for the proxy server. For example, if a transport chain includes an HTTP channel, HTTP displays in this field. If a transport chain includes both a SIP and an HTTP channel, SIP, HTTP displays in this field.

Creating a proxy cluster: Create first proxy cluster member

Use this page to specify settings for the first proxy cluster member.

There are two ways to create the first member of a proxy cluster:

- You can create the first member when you create a new proxy cluster.

  To create a new proxy cluster, in the administrative console, click **Servers > Clusters > Proxy server clusters > New**.
- You can create an empty proxy cluster and then add a first member after you finish creating the proxy cluster.

  To create a proxy cluster member for an existing proxy cluster, in the administrative console, click **Servers > Clusters > Proxy server clusters >** *proxy_cluster_name* **> Cluster members > New**.

When you create the first proxy cluster member, a copy of that member is stored as part of the proxy cluster data and becomes the template for all additional proxy cluster members that you create.

When adding servers to a proxy cluster, note that you can either remove a proxy server from a proxy cluster by deleting the proxy server from the list of proxy cluster members, or by deleting the server from the Proxy Servers collection.

Member name

Specifies the name of the proxy server that is created for the proxy cluster.

The member name must be unique on the selected node.

Select node
Specifies the node on which the proxy server resides.

Generate unique HTTP ports
Specifies that a unique HTTP port is generated for the proxy server. By generating unique HTTP ports for the proxy server, you avoid potential port collisions and configurations that are not valid.

Select basis for first proxy cluster member:
Specifies the basis you want to use for the first proxy cluster member.

- If you select **Create the member using a proxy server template**, the settings for the new proxy server are identical to the settings of the proxy server template you select from the list of available templates.
- If you select **Create the member using an existing proxy server as a template**, the settings for the new proxy server are identical to the settings of the proxy server you select from the list of existing proxy servers.
- If you select **Create the member by converting an existing proxy server**, the proxy server you select from the list of available proxy servers becomes a member of this proxy cluster.
- If you select **None. Create an empty proxy cluster** , a new proxy cluster is created but it does not contain any proxy cluster members.

Important: The basis options are available only for the first proxy cluster member. All other members of a proxy cluster are based on the proxy cluster member template, which is created from the first proxy cluster member.

Creating a proxy cluster: Create additional proxy cluster members
Use this page to create additional members for a proxy cluster. You can add a member to a proxy cluster when you create the proxy cluster or after you create the proxy cluster. A copy of the first proxy cluster member that you create is stored as part of the proxy cluster data and becomes the template for all additional proxy cluster members that you create.

To add members to a proxy cluster, in the administrative console, click **Servers > Clusters > Proxy server clusters > New >** *proxy_cluster_name* **> Cluster members > New**. After you enter the required information about the new proxy cluster member, click **Add Member** to add this member to the proxy cluster member list.

After adding a proxy cluster member, you might need to change one or more of the property settings for this proxy cluster member, or another proxy cluster member that you just added. To change one or more of the editable property settings (Member name, Node, and Generate unique ports) for any proxy cluster member that you just added, other than the first proxy cluster member, select that proxy cluster member, and then click **Edit**. When you finish changing the property settings, click **Update Member** to save your changes.

If you decide not to create a particular proxy cluster member, select the member and then click **Delete**.

You cannot edit or delete the first proxy cluster member or an already existing proxy cluster member.

If you create additional proxy cluster members immediately after you create the first proxy cluster member, the list of proxy cluster members includes a checklist in front of the names of these additional proxy cluster members. However, a check box does not appear in front of the name of the first proxy cluster member because you cannot delete this member or edit its settings. To modify the first proxy cluster member, click **Previous**.

Similarly, if you are adding proxy cluster members to a proxy cluster that already has existing members, the existing members appear in the list of proxy cluster members but a check box does not appear in front of the names of these proxy cluster members. To delete one of these existing members or to change the settings of one of these proxy cluster members, in the administrative console click **Servers > Clusters > Proxy server clusters >** *proxy_cluster_name* **> Cluster members**, and then select the member that you want to delete or whose configuration settings you want to change.

Member name

Specifies the name of the proxy server that is created for the proxy cluster.

The member name must be unique on the selected node.

Select node

Specifies the node on which the proxy server resides.

In a mixed cell environment, you can use any server from within the node group to create a new proxy cluster member. For example, if the node group to which the proxy cluster belongs consists of a V7.0 node, and a V6.1 node, you can use a server from either the V6.1 or the V7.0 node to create a new proxy cluster member.

Generate unique HTTP ports

Specifies that a unique HTTP port is generated for the proxy server. By generating unique HTTP ports for the proxy server, you avoid potential port collisions and configurations that are not valid.

Creating a proxy cluster: Summary settings

Use this administrative console page to view and save settings when you create a proxy cluster or proxy cluster member.

You can view this administrative console page whenever you create a new proxy cluster or a new proxy cluster member. This summary page displays your configuration changes before you commit the changes and the new proxy cluster or proxy cluster member is created.

To create a cluster, in the administrative console, click **Servers** > **Clusters** > **Proxy server clusters** > **New**.

To create a proxy cluster member for an existing proxy cluster, in the administrative console, click **Servers** > **> Clusters** > **Proxy server clusters** > **New** *proxy_cluster_name***Cluster members** > **New**. After you enter the required information about the new proxy cluster member, click **Add Member** to add this member to the proxy cluster member list.

The bounding node group of the proxy cluster is based on the first application server that is added as a member of the proxy cluster. The settings for this first member become the settings for the proxy cluster member template that is then used to create additional proxy cluster members.

To select a different bounding node group, in the administrative console, click **Servers** > **Clusters** > **Proxy server clusters** > **New > Cluster members > New**. Then select the appropriate node group for the new proxy cluster member. When you select a different node group, another proxy cluster member template is automatically created for that node group, if one does not already exist.

Review the changes to your configuration. Click Finish to complete and save your work.

Managing a proxy server cluster

You can use either the administrative console, or the wsadmin tool to manage a proxy server cluster .

Before you begin

The main reason for creating a proxy cluster is to make it easier to administer multiple proxy servers. For example, if your proxy servers are members of a proxy server cluster, you can specify configuration settings for that proxy server cluster, and the settings are automatically applied to all of the proxy servers in that cluster. Creating a proxy server cluster also enables you to simultaneously start and stop all of the proxy servers in that cluster.

gotcha: A proxy server cluster does not have some of the same functionality that an application server cluster has. For example, because there is no data replication among members of a proxy cluster. failover between proxy servers in the cluster is not supported. If a proxy server is down, all active connections owned by the proxy server go away and then incoming requests fail. Both proxy servers and proxy clusters, on the other hand, support the high availability and failover of backend servers, such that the proxy server can detect if a backend server is down and then forward the requests to a server that has the session replicated.

About this task

Complete one or more of the following tasks to use the administrative console to manage your proxy server clusters.

Procedure

1. In the administrative console, click **Servers > Clusters > Proxy server clusters**.
2. Select the proxy server clusters and click either **Start** or **RippleStart** to start one or more proxy server clusters.
   - **Start** launches the server process of each member of the cluster by calling the node agent for each server to start the servers. After all servers are running, the state of the cluster changes to running. If the call to a node agent for a server fails, the server does not start.
   - **RippleStart** combines stopping and starting operations. It first stops and then restarts each member of the cluster. For example, your cluster contains 3 cluster members named server_1, server_2 and server_3. When you click RippleStart, server_1 stops and restarts, then server_2 stops and restarts, and finally server_3 stops and restarts. Use the RippleStart option instead of manually stopping and then starting all of the application servers in the cluster.
3. Select the proxy server clusters and click either **Stop** or **ImmediateStop** to stop one or more proxy server clusters.
   - **Stop** halts each server in a manner that allows the server to finish existing requests and allows failover to another member of the cluster. When the stop operation begins, the cluster state changes to partially stopped. After all servers stop, the cluster state becomes stopped.
   - **ImmediateStop** stops the server quickly without regard to existing requests. The server ignores any current or pending tasks. When the stop operation begins, the cluster state changes to partially stopped. After all servers stop, the cluster state becomes stopped.
4. Click the name of the proxy server cluster to access the Configuration tab to change the configuration settings.
5. Click **New**, and type the name for the cluster in the **Cluster name** field to create a new proxy server cluster,
6. Click **Delete**, and then click **OK** to delete an existing proxy server cluster.
7. To add new members to a proxy server cluster, click the name of an existing proxy server cluster, and click **Cluster members > New**. Type the name for the proxy server in the **Member name** field.

Results

You have successfully modified a proxy server cluster.

Migrating profiles for the proxy server

This topic describes how Version 6.0.2 profiles contain the proxy server feature when migrating to Version 7.0 profiles.

About this task

Complete the following steps to migrate from a Version 6.0.2 profile to a Version 7.0 profile.

Procedure

1. Run the **WASPreUpgrade.bat** or the **WASPreUpgrade.sh** command from the `*install_root*/bin` directory and point it to the Version 6.0.2 release.

   This actions makes a copy of all the required Version 6.0.2 files that are needed for the **WASPostUpgrade.bat** or **WASPostUpgrade.sh** command. You can delete these Version 6.0.2 files, but this action is not recommended.
2. Create your corresponding profiles in Version 7.0, if you have not already done so.
3. Run the **WASPostUpgrade.bat** or the **WASPostUpgrade.sh** command from the `*install_root*/bin` directory and point the commands to the WASPreUpgrade backup directory that you created in step one.

Customizing routing to applications

This topic provides information on disabling routing through the proxy server to applications that are on on-demand configuration (ODC)-compliant application servers.

About this task

Complete these steps to disable routing to ODC-compliant application servers.

Procedure

1. From the administrative console, select **Applications > Enterprise Applications >***application_name*.
2. In the Modules section, click **Manage modules**.
3. Click a module name.
4. In the Additional Properties section, click **Web Module Proxy Configuration**.
5. In the General Properties section, deselect **Enable proxy** to disable routing using the proxy server.
6. Optional: Choose a protocol from the **Web Module Transport Protocol** list. A transport protocol, such as HTTP, can be used for the protocol between the proxy server and the application server instead of the protocol that the client uses to communicate with the proxy server.

   For example, in order to perform Secure Sockets Layer (SSL) termination, which is also known as SSL offload, for HTTPS traffic for the Web module, select **HTTP** for the transport protocol.
7. Click **Apply**.
8. Click **OK**.

Web module proxy server configuration settings

Use this page to specify proxy server configuration settings for the web module.

To view this administrative console page, click **Applications** > **Application Types** > **WebSphere enterprise applications** > ***application_name*** > **Manage Modules** > ***Web_module_name*** > **Web Module Proxy Configuration**.

Name

Specifies the name of the web module.

Enable Proxy

Select **Enable Proxy** to enable proxy server to route requests to this web module. Deselect **Enable Proxy** to disable routing using the proxy server.

Web Module Transport Protocol

Specifies the protocol that the proxy server uses when communicating with this web module.

Choose the protocol in the web module transport protocol field if you want to use a specific transport protocol, such as HTTP, for the protocol between the proxy server and the application server. This is an alternative to using the protocol that is used by the client to communicate with the proxy server. For example, in order to perform Secure Sockets Layer (SSL) termination, which is also known as SSL offload, for HTTPS traffic for the web module, select HTTP for the transport protocol. HTTPS protocol is for advanced configurations and is not commonly used.

Custom Properties

Specifies additional custom properties for this runtime component. Some components use custom configuration properties that are defined by this option.

Routing requests to ODC-compliant application servers in other cells

On Demand Configuration (ODC) enables product components, such as the proxy server, to build application deployment, availability, and accessibility information in order to route requests for these applications without any additional configuration at the proxy server. If you want to isolate proxy servers with firewalls, place them in a cell that is separate from the applications and configure the core group bridge service.

About this task

ODC uses the high availability capabilities of the product to publish and subscribe updates. The deployment manager within a cell and the application servers that have the ODC capabilities typically publish the updates that are subscribed to by intermediaries such as the proxy server.

You can configure the proxy server to route requests to applications that are hosted in other cells. The core group bridge service provides communication between a cell with a proxy server and a cell with a target application. Once the cells are linked with core group bridges, the proxy server will be able to route to the application without additional configuration. To view your core group bridge settings, in the administrative console, click **Servers > Core groups > Core group bridge settings**.

Note: Cross-cell failover is not supported. If the same application is installed on both the local cell and in one or more remote cells, and the application or application servers in the local cell go down, then the ones in the remote cell will not be used. The proxy server will always route to the local cell even if the resource is available in a remote cell.

The basic steps to configure cross-cell routing are configuring and enabling core group bridges in the cell that the proxy server belongs to, and in each of the other cells that are being routed to. The core group bridges need to be configured with the same access point group name.

Procedure

1. Create bridge interfaces, peer access groups and peer ports for cells. The peer access point group name must be the same in the cells.
2. Restart all processes that are now bridge interfaces.

What to do next

If security is enabled, it must be enabled in all cells for the core group bridge service.

Configuring rules to route requests to web servers

This section provides information to configure rules to route requests to web servers or application servers that are not on-demand configuration (ODC)-compliant.

About this task

The proxy server permits explicit configuration of routing rules in order for client requests to route to web servers or application servers that are not ODC-compliant. This involves completing following tasks.

Procedure

1. Create a URI group.

   To define the URI patterns that are associated with the web content that is hosted on these servers, in the administrative console click **Environment > URI groups**.
2. Create a generic server cluster definition.

   To define the endpoints that define the cluster membership, in the administrative console click **Servers > Clusters > Generic server clusters**.
3. Create routing rules from the detail view of the proxy server panel that associates the inbound virtual host and a defined URI group to a defined generic server cluster.

Modifying the HTTP endpoints that the proxy server listens on

By default, the proxy server listens on the following named endpoints: PROXY_HTTP_ADDRESS and PROXY_HTTPS_ADDRESS. You can modify the ports and hosts that are associated with these endpoints in the administrative console.

About this task

Modify the ports and hosts that are associated with the proxy server endpoints as follows:

Procedure

1. In the administrative console, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> Communications > Ports**.
2. Select the port you want to modify.

   The host and port that are associated with the specified endpoints can be modified to suit the requirements of the deployment. The administrator must ensure for correct routing, the virtual hosts that are associated with applications to be routed to are correctly updated with the appropriate host aliases (host aliases with the modified host and port combinations).
3. Click **Apply**.

What to do next

Restart the server. The server must be restarted for this modification to go into effect.

Adding a new HTTP endpoint for the proxy server

An administrator can use the proxy server to create additional endpoints to listen for HTTP and HTTPS requests.

About this task

The following steps will create a new transport chain:

Procedure

1. Click **Servers** > **Server Types** > **WebSphere proxy servers** > *proxy_server_name* > **HTTP proxy server settings** > **Proxy server transports** in the administrative console.
2. Click **New** to launch the Create new transport chain wizard.
3. Determine a name for the transport chain.
4. Choose a template based on whether the endpoint should accept HTTP (proxy template) or HTTPS (proxy-secure template) requests. The wizard prompts for the host and port for the endpoint. Creating a new endpoint for the proxy server requires a restart of the proxy server to be effective.

Setting up caching in the proxy server

This topic provides information on caching static and dynamic content in the proxy server.

About this task

Complete the following steps to configure a proxy server such that it can cache static and dynamic content.

Procedure

1. Configure the object cache instance for size, disk offload location, and other such capabilities, in the administrative console. Click **Servers** > **Server Types** > **WebSphere proxy servers** > *proxy_server_name* > **HTTP proxy server settings** > **Proxy cache instance config**. Repeat these steps on any nodes that have a proxy server.
2. Select the proxy cache store instance and enable configuration attributes such as cache size, disk offload, and cache replication. For disk offload, it is recommended that the location be set to a dedicated disk partition.
3. Enable caching at the proxy server, in the administrative console. Click **Servers** > **Server Types** > **WebSphere proxy servers** > *proxy_server_name* > **HTTP proxy server settings** > **Proxy settings** page in the administrative console.
4. Select **Enable caching** and choose a cache instance from the drop-down box.
   a. To enable dynamic content to be cacheable with the proxy server, in the administrative console, click **Servers** > **Server Types** > **WebSphere proxy servers** > *proxy_server_name* > **HTTP proxy server settings** > **Proxy settings**, and then select **Cache dynamic content**. You enable cacheablity and invalidation of dynamic content when you enable servlet caching on the application server, and specifying the cache criteria in a `cachespec.xml` file that is associated with that application. Invalidations are received by connecting to the cache update URI that is associated with the invalidation servlet hosted on the application server cluster.

      Dynamic content is content that an application, that is hosted on an application server, generates. A proxy server caches dynamic content only if the content is identified as edge cacheable in the `cachespec.xml` file for the application. All of the information that describes the cache, such as the ID to use for the cache, dependency identifiers for invalidation, and expiration times, is also defined in the `cachespec.xml` file. Proxy Server uses the ESI protocol to obtain this information from the file.

      See the *Administering applications and their environment* PDF for more information on how to set up a `cachespec.xml` file for an application.

      Cached dynamic content can be invalidated by events in the application server. The ESI Invalidation Servlet, that is contained in the DynacacheEsi.ear application, propagates these invalidation events from the application server to the proxy server. The DynacacheEsi.ear is shipped with the product, and must be deployed in the cluster with the application that is generating the dynamic content for dynamic caching at the proxy server to function properly.

   b. Static caching is enabled by default when caching is enabled for the proxy server. *Static content* is web content that is public and accompanied by HTTP response headers, such as EXPIRES and LAST_MODIFIED_TIME, that describe how long the response can be cached. The proxy server uses the HTTP 1.1 RFC (2616), which specifies how content should be treated and includes capabilities such as VARY header support for caching variants of the same resource Uniform Resource Identifier (URI).

Static cache rules collection

This topic lists the static cache rules for a proxy server. From this topic you can create, delete, or modify a static cache rule.

To view this administrative console topic, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> HTTP Proxy Server Settings > Static cache rules**.

The security proxy, or the intermediary that is the entry point into the enterprise, is responsible for terminating SSL connections with the client, authenticating the request, propagating the connection characteristics for the client, and any other credentials to the application server in the enterprise.

The proxy server enables security proxies to be identified so that private headers that are set in the request are propagated as they are to the application servers. Identify the list of security proxies that are trusted by the proxy server using the trusted security proxies field. To access this administrative console field, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> HTTP proxy server settings > Proxy settings**. You can find trusted security proxies in the **Security** section of this administrative console page.

URI Groups

The URI group name is a user-specified name.

The name field cannot contain the following characters: `# \ / , : ; " * ? < > | = + & % '`

The name that is defined must be unique among URI groups and cannot begin with a period or a space. A space does not generate an error, but leading and trailing spaces are automatically deleted.

Disable caching for this URI group

Specifies whether or not caching is disabled.

The default is `false`, which indicates that caching is enabled for the URI group.

Default expiration

The default expiration that you set for the cached response for the URI that is associated with this cache rule.

The default expiration value is in seconds.

Last modified factor

Use this field to derive the cache expiration value for a response if it does not have HTTP expiration headers and when it has a LastModifiedTime header in the response.

Name of the virtual host

A virtual host that is configured using the virtual host service. This virtual host is associated with the proxy server. This attribute is one of the elements in a request that is matched by the proxy server to determine if this rule is activated.

Static cache rule settings

Use this topic to configure a cache rule that is associated to a URI group for the proxy server. HTTP 1.1 defines a set of rules for proxy servers to cache content. Static cache rules enable these default rules to be overridden for a given address space. Before the rules have any meaning, you must enable caching on the **Servers** > **Server Types** > **WebSphere proxy servers** > *proxy_server_name* > **HTTP Proxy Server Settings** > **Proxy settings** administrative console page.

To view this administrative console page, click **Servers** > **Proxy Servers** > *server_name* > **HTTP Proxy Server Settings** > **Static cache rules** > *URI_group*.

You can edit proxy server setting fields on the Configuration tab.

URI groups

URI groups, along with the virtual host, define the scope of the address space to have cache customizations performed.

The name field cannot contain the following characters: **# \ / , : ; " * ? < > | = + & % '**

The name that is defined must be unique among URI groups and cannot begin with a period or a space. A space does not generate an error, but leading and trailing spaces are automatically deleted.

Disable caching for this URI group

Disables caching for this address space. A user may wish to disable caching for a set of content servers that are known to contain sensitive or highly-personalized information.

The default is `false`, which indicates that caching is enabled for the URI group.

Default expiration

The default expiration value, in seconds, that is used to determine the validity of a cached response when all other HTTP 1.1 caching-related response headers do not give guidance. The default value is sufficient in most environments.

The default expiration value is in seconds.

Last modified factor

The percentage of a last-modified header for a response that determines the validity of a cached response when the response does not have explicit HTTP expiration headers. The default value is sufficient in most environments.

The default for this field is `0.0`.

Name of the virtual host

A virtual host that is configured using the virtual host service. This virtual host is associated with the proxy server. This attribute is one of the elements in a request that is matched by the proxy server to determine if this rule is activated.

The default for this field is `none`.

Routing requests from a plug-in to a proxy server

This topic provides information on setting up a web server plug-in to route requests to a proxy server.

About this task

An administrator might choose to set up a web server, such as IBM HTTP Server, with the web server plug-in as a front-end to the proxy server. The plug-in configuration file for such a topology cannot use the traditional plug-in configuration generation mechanism if the requests are routed through the proxy server.

To generate the `plugin-cfg.xml` file to use with the web server plug-in to route through the proxy server, complete the following steps:

Procedure

1. From the administrative console, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> HTTP proxy server settings > Proxy settings**.
2. In the **Generate plug-in configuration** drop-down menu, select the appropriate scope.
3. Optional: If you have a script that manually copies the `plugin-cfg.xml` file from the node to the plug-in installation location, enter the path to the script in the **Plugin config change script** field.
4. In the Trusted Security Proxy field, add the hostname or IP address of the node for the plug-in that serves as the trusted intermediary for the proxy server.
5. Click **OK**.
6. Disable the automatic propagation of the plug-in if you are using IBM HTTP Server with remote administration. From the administrative console, click **Servers > Server Types > Web Servers >** *web_server_name* **> Plug-in properties**. Deselect **Automatically propagate plugin configuration file**. This will prevent WebSphere Application Server from copying the traditional `plugin-cfg.xml` file over the proxy server `plugin-cfg.xml` file.
7. Save your changes.
8. Stop and restart the proxy server. The `plugin-cfg.xml` file will be in the `{WAS_ROOT}/profiles/`*`profilename`*`/etc` directory unless your plug-in was generated for the server scope. If you generated the plug-in for the server scope, the `plugin-cfg.xml` file will be in the `{WAS_ROOT}/profiles/`*`profilename`*`/etc/`*`server_name`* directory. If you do not have a script in the **Plugin config change script** field, manually copy the `plugin-cfg.xml` file to the plug-in.

   gotcha: During the plugin-cfg.xml generation process temporary plugin-cfg-*xxxx*.xml files are created. If you notice any of these temporary files in the same directory as the plugin-cfg.xml file, you can either ignore them, or manually delete them.

Results

After a plugin-cfg.xml file is initially generated, it is automatically regenerated whenever the On Demand Configuration (ODC) changes, For example, the plugin-cfg.xml file is regenerated whenever an application is installed or uninstalled, or the weights assigned to one or more proxy server cluster members changes.

What to do next

To verify that the proxy server trusts the web server, add the host name or address of the web server to the Trusted security proxies section on the Proxy Settings page in the administrative console. To reach this page, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> HTTP proxy server settings > Proxy settings** This enables the proxy server to honor the private headers that are set by the fronting intermediary server.

Creating a proxy server cluster using the wsadmin command

If you run multiple proxy servers on your system, you might want to create a proxy server cluster to make it easier to administer these proxy servers. You can use either the administrative console or the **wsadmin** command to create a proxy server cluster.

Before you begin

A proxy server cluster includes the machines, and nodes that are going to belong to the proxy server cluster. Before you define your proxy server cluster:

- Determine whether a domain name server (DNS), a load balancer, or a proxy server is going to be used to route requests to the proxy server cluster.
- Verify that Version 7.0 of the product is installed on the nodes that will belong to the proxy server cluster.

gotcha: A proxy server cluster does not have some of the same functionality that an application server cluster has. For example, a proxy cluster does not provide failover, or high availability support. The main reason for creating a proxy cluster is to make it easier to administer multiple proxy servers. For example, if your proxy servers are members of a proxy server cluster, you can specify configuration settings for that proxy server cluster, and the settings are automatically applied to all of the proxy servers in that cluster. Creating a proxy server cluster also enables you to simultaneously start and stop all of the proxy servers in that cluster.

About this task

Complete the following tasks to create a proxy server cluster.

Procedure

1. Start the deployment manager.
2. Start the wsadmin utility.
3. Issue one of the following commands.

   The following command creates an empty cluster with no members:

   ```
   $AdminTask createCluster {-clusterConfig {-clusterName <name_of_cluster> -clusterType PROXY_SERVER}}
   ```

   The following command creates a cluster, and add a proxy server to that cluster:

   ```
   $AdminTask createCluster { -clusterConfig {{<name_of_cluster> true PROXY_SERVER}} -convertServer
        {{<node_name> <name_of_proxy_server> "" "" ""}}}
   ```

   The proxy server that you add is used as the template for all subsequent proxy servers that you add to the cluster.
4. Add a member to the cluster:

   If no members exist in the cluster, the first proxy server that is added serves as the template for subsequent members that you add to the cluster.

   Issue the following command to add a member to the cluster:

   ```
   $AdminTask createClusterMember {-clusterName <name_of_cluster> -memberConfig
         {-memberNode <node_name> -memberName <name_of_proxy_server>}}
   ```

   When a proxy server is added to a cluster, proxy-specific configuration settings for it can only be configured using the wsadmin scripting client.
5. Issue the `$AdminConfig save` command to save the configuration changes.
6. Start the proxy server cluster.

   Start the proxy server cluster automatically enables request routing.
7. Configure requests to route to the proxy server.

   For DNS-based routing, associate the logical name of the site with the IP addresses of the proxy server cluster members in DNS.

   For Load balancer routing, configure the IP addresses of the cluster members as the target of the virtual cluster.

   For Edge proxy or IBM HTTP Server with WebSphere Application Server plug-in-based routing, generate the plug-in configuration file for the proxy server cluster, and configure the Edge proxy or the WebSphere Application Server plug-in with this information.

Results

The proxy server cluster is created with the members and is enabled for routing traffic.

What to do next

Monitor the traffic.

Monitoring the proxy server with PMI

You can monitor the traffic of a proxy server using the Performance Monitoring Infrastructure (PMI) function.

Before you begin

The proxy server must be running before performing these steps.

Procedure

1. In the administrative console, click **Monitoring and Tuning > Performance Monitoring Infrastructure (PMI)** in the console navigation tree. The proxy server collection page displays.
2. Select the proxy server from the collection list.
3. Click **Custom**. The Custom monitoring level panel displays.
4. In the navigation tree, expand **Proxy Module**. Select the statistics you want to view, then click **Enable**.

Monitoring traffic through the proxy server

You can monitor traffic, such as requests and connection statistics, through the proxy server.

Before you begin

You should know the machines and nodes that will belong to the proxy server cluster before completing these steps, because the product must be installed on those machines.

Procedure

1. Ensure that the proxy server is running and there is some traffic flowing through the proxy server.
2. Obtain the proxy server MBean and invoke the operation to get the route statistics as follows:
   a. Start the wsadmin scripting tool.
   b. Get all of the traffic statistics through the proxy server using one of the following commands.

      Using the JACL command

      ```
      $AdminControl queryNames type=ProxyServer,*
      set proxymbean <cut and paste the MBean identifier from the previous command output>
      $AdminControl getAttribute $proxymbean stats
      ```

      Using the Jython command

      ```
      print AdminControl.getAttribute((AdminControl.queryNames("type=ProxyServer,*")), "stats")
      ```

      The **$AdminControl queryName** command lists all of the proxy server MBeans. There will be one per active proxy server in the cell. Set the *proxymbean* variable to the appropriate proxy server MBean from the output of the previous command.

Results

The traffic that flows through the proxy server can now be monitored.

Overview of the custom error page policy

The custom error page policy is a feature that enables the proxy server to use an application to generate an HTTP error response. With this capability, the administrator can return a polished error page when the proxy server generates an error, or when a content server returns an unsuccessful response.

The following action describes scenarios for how the error page policy is used when it is configured:

- **Internal error**

1. The client sends the following request to the proxy server: `GET /house/rooms/kitchen.jpg HTTP/1.1`.
2. The proxy server generates an internal error because no servers map to the request (HTTP 404 – File not found).
3. The error policy is configured to handle HTTP 404 responses, so it sends a request to the error page application to retrieve error content to send to the client. The request URI and HTTP response code are included as query parameters in the request to the error page application. If the configured error page application URI is /ErrorPageApp/ErrorPage, then the request URI to the error page application is: `/ErrorPageApp/ErrorPage?responseCode=404&uri=/house/rooms/kitchen.jpg`. The query parameters "responseCode" and "uri" are sent to the error page application by default.
4. The proxy server returns the response code and the content that the error page application returned. The error page application can also be set up to return the response code that is passed in the responseCode query parameter.

- **Remote error**
  1. The client sends the following request to the proxy server: `GET /house/rooms/kitchen.jpg HTTP/1.1`
  2. The proxy server forwards the request to the `homeserver.companyx.com` content server.
  3. The `homeserver.companyx.com` content server is unable to locate the `/house/rooms/kitchen.jpg` file and sends an HTTP 404 response (File not found) to the proxy server.
  4. The error policy is configured to handle HTTP 404 responses, so it sends a request to the error page application to retrieve error content to send to the client. The request URI and HTTP response code are included as query parameters in the request to the error page application. If the configured error page application URI is `/ErrorPageApp/ErrorPage`, then the request URI to the error page application is: `/ErrorPageApp/ErrorPage?responseCode=404&uri=/house/rooms/kitchen.jpg`. The query parameters "responseCode" and "uri" are sent to the error page application by default.
  5. The proxy server returns the response code and the content that the error page application returned. The error page application can also be set up to return the response code that is passed in the responseCode query parameter.

A sample error application is available in the `<WAS_INSTALL_ROOT>/installableApps/HttpErrorHandler.ear` file.

Request mapping

This topic provides an overview of how the proxy server matches an HTTP request that is received to an application that is deployed in the cell or routing rule.

Unlike the Apache Web Server or Caching Proxy, which have flat configuration files with routing precedence that is inherent to the ordering of directives, the proxy server uses a best match mechanism to determine the installed application or routing rule that corresponds to a request. The virtual host or URI patterns determine the best match for a web module or routing rule. For applications that are deployed in clusters, the proxy server maintains affinity (Secure Sockets Layer ID, cookie, and URL rewriting), otherwise, a weighted round-robin approach is used to select the target server. The following examples address various routing scenarios for when routing rules and applications are deployed within the same cell.

Proxy environment

A WebSphere proxy server called `proxy1` is active in the same cell as the applications and routing rules. All of the applications and routing rules are enabled in the cell for `proxy1`, and PROXY_HTTP_ADDRESS for proxy1 is set to `80`.

Virtual host	Host name	Port
default_host	host1.company.com	80
	host1.company.com	9080

Virtual host	Host name	Port
	*	80
proxy_host	host2.company.com	80
	*	443
	*	80
server_host	host3.company.com	80

URI group name	URI patterns
ALL	/*
ROOMS	/kitchen/*, /bathroom/*, /bedroom/*
CONFLICT	/WM2C/*

Generic server cluster name	Protocol	Host	Port
CLUSTER1	HTTP	webserver1.company.com	9081
		webserver2.company.com	9083
CLUSTER2	HTTP	host47.company.com	8088
		host48.company.com	8088
CLUSTER2-SSL	HTTPS	host47.company.com	8443
		host48.company.com	8443

Routing rule name	Virtual host	URI group	Action
ALLTOCLUSTER1	proxy_host	ALL	Generic server cluster - CLUSTER1
ROOMTOCLUSTER2	proxy_host	ROOMS	Generic server cluster - CLUSTER2
ALLTOCLUSTER2	server_host	ALL	Generic server cluster - CLUSTER2
REDIRECTTOCONFLICT	default_host	CONFLICT	Redirect - http://www.conflict.com

Application name	Context root	Web module name	Virtual host	Web module URI patterns
App1	/WM1A/	Web Mod A	default_host	wm1a.jsp
	/WM1B/	Web Mod B	default_host	wm1b.jsp
App2	/WM2C/	Web Mod C	default_host	/*, wm2c.jsp
	/WM2D/	Web Mod D	default_host	/*, wm2d.jsp

Example 1: Basic request

The `proxy1` proxy receives the following request:

```
GET /WM1A/wm1a.jsp HTTP/1.1
Host: host1.company.com
```

Result

The `wm1a.jsp` file is sent as the response. The ALLTOCLUSTER1 routing rule is a possible match, but Web Mod A is chosen as the best match by proxy1 because the combination of its context root and URI pattern `/WM1A/wm1a.jsp` is a better match than `/*`. Web Mod A is also chosen as the best match because its virtual host contains the `host1.company.com:80` alias, which is a more specific match than the `*:80` wild card alias.

Example 2: Routing rules that use the same URI group and different virtual hosts

The `proxy1` proxy receives the following request:

```
GET /index.html HTTP/1.1
Host: host3.company.com
```

Result

The `proxy1` proxy maps the request to the ALLTOCLUSTER2 routing rule, and a response is received from a server in CLUSTER2. The ALLTOCLUSTER1 routing rule is a possible match and can handle the request if the ALLTOCLUSTER2 routing rule did not exist. However, the ALLTOCLUSTER2 rule is the best match because its virtual host (server_host) explicitly lists `host3.company.com`.

Example 3: Routing rules that use same virtual host and different URI groups

The `proxy1` proxy receives the following request:

```
GET /kitchen/sink.gif HTTP/1.1
Host: host2.company.com
```

Result

The `proxy1` proxy maps the request to the ROOMSTOCLUSTER2 routing rule and a server from the CLUSTER2 cluster sends a response. The ALLTOCLUSTER1 routing rule is a possible match, but the ROOMSTOCLUSTER2 rule is the best match because its URI group contains a pattern `/kitchen/*` that is a better match for the request URI `/kitchen/sink.gif`.

Example 4: Routing rule URI group conflicts with URI pattern of a web module that uses the same virtual host

The `proxy1` proxy receives the following request:

```
GET /WM2C/index.html HTTP/1.1
Host: host1.company.com
```

Result

The result is indeterminate. It is unknown whether Web Mod C or the REDIRECTTOCONFLICT routing rule handles the request because they use the same virtual host and have the same URI pattern. In such cases, the ID DWCT0007E message is displayed in `SystemOut.log` file for the `proxy1` proxy. In this example, changing the REDIRECTTOCONFLICT routing rule to use a different virtual host resolves the problem.

Note: This topic references one or more of the application server log files. Beginning in WebSphere Application Server Version 8.0 you can configure the server to use the High Performance Extensible Logging (HPEL) log and trace infrastructure instead of using `SystemOut.log` , `SystemErr.log`, `trace.log`, and `activity.log` files or native z/OS logging facilities. If you are using HPEL, you can access all of your log and trace information using the LogViewer command-line tool from your server profile bin directory. See the information about using HPEL to troubleshoot applications for more information on using HPEL.

Example 5: The PROXY_HTTP_ADDRESS address is not in the virtual host

Assume that the `proxy1` proxy address, PROXY_HTTP_ADDRESS, is changed to `81`, while all of the other configuration information remains the same. The `proxy1` proxy receives the following request:

```
GET /index.html HTTP/1.1
Host: host1.company.com:81
```

Result

The `proxy1` proxy is unable to handle the request because the PROXY_HTTP_ADDRESS address is not available in a virtual host and will send an HTTP 404 response back to the client.

Session failover in the proxy server

This article describes how the proxy server handles session failover.

You can enable memory-to-memory replication on an application server to maintain session state in multiple servers. In this case, a private header is added to the response which identifies the backup servers for the session. The proxy server reads this header and maintains a list of backup servers for a session. If the proxy server fails to route to the primary server, it tries to route to the backup servers. If none of the backup servers are available, a deterministic algorithm selects one of the available servers; consequently, multiple proxy servers will route to the same server.

If the set of servers that are hosting a session changes, the private response header causes the proxy server to update its list of servers for the session. It is possible that the set of servers is updated, but the proxy server has not yet received an updated response header. In this case, the proxy server routes to a server that does not contain the session data. If this occurs, the backend server obtains the session data from a server that contains the session data. There is no functional difference in this case; however, there is a performance difference due to the cost of obtaining the session data from another server.

Installing a Session Initiation Protocol proxy server

The Session Initiation Protocol (SIP) proxy server initiates communication and data sessions between users. It delivers a high performance SIP proxy capability that you can use at the edge of the network to route, load balance, and improve response times for SIP dialogs to backend SIP resources. The SIP proxy provides a mechanism for other components to extend the base function and support additional deployment scenarios. This topic provides information to install a Session Initiation Protocol (SIP) proxy server.

Before you begin

Ensure that you have a SIP application installed. The SIP container will not listen on a port without a SIP application installed. For a proxy server to be useful, you should have a cluster of SIP application servers.

About this task

The SIP proxy design is based on the HTTP proxy architecture and can be considered a peer to the HTTP proxy. Both the SIP and the HTTP proxy are designed to run within the same proxy server and both rely on a similar filter-based architecture for message processing and routing.

A SIP proxy serves as the initial point of entry, after the firewall, for SIP messages that flow into and out of the enterprise. The SIP proxy acts as a surrogate for SIP application servers within the enterprise. In fact, the nodes that host the SIP proxy servers host the public SIP domain of the enterprise. As a surrogate, you can configure the SIP proxy with rules to route to and load balance the clusters of SIP containers. The SIP proxy is capable of securing the transport, using secure sockets layer (SSL), and the content, using various authentication and authorization schemes.

The SIP proxy is also responsible for establishing outbound connections to remote domains on behalf of the back-end SIP containers and clients that reside within the domain that is hosted by the proxy. Another important feature of the SIP proxy is its capability to protect the identity of the back-end SIP containers from the SIP clients.

gotcha: You must first have created a cell, deployment manager, and node when you installed WebSphere Application Server.

Procedure

Create a proxy server on the node that you just federated, or on another node in the cell that contains the cluster that the proxy will sit in front of by clicking **Servers > Server Types > WebSphere proxy servers > New**.

gotcha: You must select the SIP protocol.
You do not have to install the proxy server on a machine that also includes one of the servers in the cluster.

Results

After you choose a default cluster, your SIP proxy server will be functional. However, if the SIP proxy server is fronted by an IP sprayer, you must set up the loopback address and modify the inbound channel chains to make the SIP proxy server functional. See the load balancer documentation for information about setting up the loopback address.

gotcha: The virtual host for your SIP container ports must be defined, and the SIP container(s) must be restarted after adding the new virtual host.

Trusting SIP messages from external domains

The general approach for providing secure communications between two independent domains or communities (each maintaining distinct directories) relies on *identity assertion*, where a trust relationship is established between two distinct domains using a certificate exchange during the setup of the physical Secure Sockets Layer (SSL) connection between the two domains.

About this task

Authentication of Session Initiation Protocol (SIP) messages that are sent by end users needs to occur only in the local domain for the user. All user messages pass through the SIP container local domain before being sent on to the external domain. If a message is received from a external domain over a secured connection that is mutually authenticated in the manner described as follows, it is assumed that the message is authenticated by the external domain because of the trust relationship. An administrator can enable support for external domains in the SIP proxy as follows:

Procedure

1. Enable client authentication within the SSL repertoire that is assigned to all the inbound channel chains (or endpoints) that are to receive inbound connections from external domains.
2. Ensure that all trusted certificate authorities are set up in the trust store that is assigned to the SSL repertoires mentioned in the previous step. Set up the asymmetric key pair (public and private keys) for the local domain, with the proper chain of certificates that is associated with the local domain.
3. Configure the distinguished names (DNs) that are associated with the external domains to support. The DN is part of the X.509 certificate that is sent by the external domain server when the SSL connection is set up. Within the configuration model, each SIP external domain entry includes a field for the external DN.
4. Assuming that the SIP infrastructure is deployed within each domain, provide the DN to the external domain administrator that is included in the local domain's public certificate. With this action, the external domain administrator can configure the proper external DN.

   With this approach, the Java Secure Socket Extension (JSSE) is responsible for authorizing the certificate that is received over a new inbound connection from a external domain. This authorization is based on the agreed upon certificate authorities whose certificates are set up in the local trust store. If the external domain certificate is authorized, it is then the responsibility of the SIP proxy to filter the connections, based on the DN that is associated with the external domain certificate. The proxy also validates outbound connections by ensuring that the DN that is received in the remote server certificate matches the DN configured for the external domain.

The SIP proxy must recognize when identity assertion is in use so that it can inform the SIP container that no message authentication is required over this mutually authenticated connection. This communication is done by adding the P-Preferred-Identity SIP header, which is described in RFC 3325, in all SIP messages that are sent from the proxy to the SIP container that arrive over the authenticated connection. The SIP container only recognizes this header when it is received from a device that resides in the trusted domain, specifically the SIP proxy. It is up to the SIP proxy to remove this header from any inbound messages that are received over any connections to remote devices that are not considered part of the trusted domain. You can also use this header to support the addition of proxy authentication.

Tracing a Session Initiation Protocol proxy server

You can trace a Session Initiation Protocol (SIP) proxy server, starting either immediately or after the next server startup.

About this task

By default, the trace messages do not include the Authorization and Proxy-Authorization headers. If you also do not want the other SIP message headers to appear in the trace log, add the SIP UDP transport channel custom property `hideMessageHeaders` custom property to the settings for the SIP inbound channel (SIP 1) and set the property to `true`. You can also add the `hideMessageBody` custom property to the settings for this channel and set it to `true` if you do not want the message body included in the trace messages.

Note: This topic references one or more of the application server log files. Beginning in WebSphere Application Server Version 8.0 you can configure the server to use the High Performance Extensible Logging (HPEL) log and trace infrastructure instead of using `SystemOut.log` , `SystemErr.log`, `trace.log`, and `activity.log` files or native z/OS logging facilities. If you are using HPEL, you can access all of your log and trace information using the LogViewer command-line tool from your server profile bin directory. See the information about using HPEL to troubleshoot applications for more information on using HPEL.

To trace a SIP proxy server, complete the following steps:

Procedure

1. Start the product, and open the administrative console.
2. In the administrative console, click **Troubleshooting** > **Logs and trace**.
3. Select the name of the server for the SIP proxy server.
4. From the General Properties section, click **Diagnostic Trace Service**.
5. Select one of the following options:

Option	Description
Configuration	To start tracing after the next server startup
Runtime	To start tracing immediately

6. Replace the content of the trace specification with the following code: `*=info:com.ibm.ws.sip.*=all:com.ibm.ws.proxy.*=all`.
7. Make sure that the **Enable trace with following specification** check box is checked.
8. Click **Apply** > **Save**.

What to do next

When the changes take effect, SIP proxy server tracing messages display in `WASProductDir/logs/serverName/trace.log` on the SIP proxy server node, where *WASProductDir* is the fully-qualified path name of the directory where the product is installed and *serverName* is the name of the specific instance of the application server that is running the SIP proxy server to be traced.

High availability and workload management with Session Initiation Protocol proxy server

The Session Initiation Protocol (SIP) high availability solution assumes that all messages that belong to the same dialog are handled by the same container. If a container fails, all of the sessions that were handled by that container are picked up by the other servers in that container replication domain, and are activated immediately. All subsequent messages that belong to a session from the failed container are sent to the new container in charge of that session.

gotcha: The SIP proxy server does not administer transaction-level failover, which are calls that are made in the middle of a transaction. The SIP proxy server administers failover of stable calls, which are calls that are not made in the middle of a transaction.

High availability manages the following:

- Scalability – The ability to add more servers to the cluster to handle increased loads.
- Load balancing – The ability to distribute the load across all of the servers in the cluster so no server is overloaded while there are other servers that are not utilized.
- Fail over – The ability to recover from a failure in one or more of the components in the solution.

The SIP high availability solution uses the following components:

- SIP container - Maintains all of the sessions and launches all of the applications.
- SIP proxy server - Manages a large number of client connections, routes incoming messages to the appropriate SIP container, and creates outbound connections to clients and other domains.
- Network Dispatcher - Provides a single IP for the cluster and round-robins between proxies.
- Unified Clustering Framework (UCF) - Communicates routing information between the SIP container and the SIP proxy. Using UCF, the SIP proxy routes messages to the least-loaded SIP container or to a container that is taking over sessions for a failed server.

  Note: If you add SIP containers to a cluster while traffic is flowing, you should add them one container at a time so that the system can go through the bootstrap process for the container without draining resources from the entire cluster. If you add one container at a time, only the added container goes through the bootstrap process as opposed to all of the containers in the cluster.

Manage failover in the SIP proxy by:

1. Replicating the data in the sessions between SIP containers so that other containers are able to activate the failed sessions in case of a server failure.
2. Activating the failed sessions on the rest of the servers immediately when a failure is detected because the SIP sessions might have timers associated with them.
3. Routing incoming messages that belong to failed sessions to the new server that is handling the session.

Load balancing with the Session Initiation Protocol proxy server

Load Balancer for IBM® WebSphere® Application Server can help maximize the potential of your Web site by providing a powerful, flexible, and scalable solution to peak-demand problems. Configuration of a load balancer is critical to prevent any single points of failure at Session Initiation Protocol (SIP) proxy. Configuration of a load balancer is required when more than one SIP proxy is deployed.

Before you begin

Before you begin, complete these tasks:

1. Install Load Balancer for IBM WebSphere Application Server. See the information provided in the Edge Components Information Center. This information center is available on the WebSphere Application Server Library page.
2. From the Load Balancer for IBM WebSphere Application Server, complete the steps for setting up server machines for load balancing. Ensure that you set up a loopback address for your operating system. See the Load Balancer Administration Guide in the Edge Components Information Center.
3. Start the Session Initiation Protocol (SIP) proxy server.

About this task

Complete these steps to integrate the SIP proxy server with the Load Balancer.

Procedure

1. Start the Load Balancer.
   a. From the command prompt, type `dsserver start`
   b. Then type `lbadmin` to start the administrative console for the Load Balancer.
   c. From the administrative console, right click **Dispatcher**, and then select **Connect to Host**
   d. Right click the host name, and select **Start Executor**.
2. Start the configuration wizard for the Load Balancer.
   a. Select default host.
   b. Type a cluster address. You cannot ping the cluster address before the Executor starts.
   c. Define your SIP traffic port. Type a port number, such as 5060. You must specify this same value for port when you create a user-defined port in on your proxy server.
   d. Add each server to which the Load Balancer proxies traffic. In your configuration, the load balanced server is the proxy server for your configuration.
   e. Start an advisor by typing the name of the advisor. For example, start the HTTP advisor for HTTP traffic. For SIP traffic, start the SIP advisor. The advisor tells the manager if a specific port is accepting traffic.
3. Set up a loopback address on the physical proxy server host that represents the Virtual IP address for the cluster that is defined on the load balancer. Refer to the topic on configuring the server machines in the information center for Load Balancer.
4. Configure an IP sprayer from the application server's administrative console
   a. From the administrative console, click **Servers** > **Server Types** > **WebSphere proxy servers** > ***proxy_server_name*** > **SIP proxy server settings** > **SIP proxy settings**.
   b. Scroll down to the health checking setting for Load Balancer, and specify the physical IP address of the host on which Load Balancer is installed.
   c. Specify the SIP health check method name. The IBM Load Balancer uses OPTIONS as its method name.
   d. Click on **Apply**, and save your actions.
5. Optional: From the administrative console, define custom properties for the SIP proxy server. These custom properties can be used for defining SIP Proxy Load Balancer Health checking.
   a. In the Additional Properties section, click **Custom properties**.
   b. Ensure that Proxy servers have both the LBIPAddr and SIPAdvisorMethodName custom properties defined. The information for these custom properties is:

Name	Value
LBIPAddr	Physical IP address of the machine where the IBM Load Balancer is installed
SIPAdvisorMethodName	OPTIONS

Specify a value for these SIP proxy custom properties, and click **OK**.

Attention: These settings are now available from the SIP proxy settings page in the administrative console, and they do not require that a custom property be set, but defining the custom properties is supported. If the custom property is set, the value of the custom property takes precedence over the value that you set in the administrative console.

6. Create a user-defined port from the application server's administrative console.
   a. From the administrative console, click **Servers** > **Server Types** > **WebSphere proxy servers** > ***proxy_server_name*** > **Ports** > **New**.
   b. Select **User-defined port**.
   c. Enter `SIP_LB_Address` for the Port Name field.
   d. Enter a value for the Host field. This is the virtual IP (cluster address) that is configured on your load balancer.
   e. Enter a value for the Port field. This value corresponds to the port through which the load balanced servers are configured to accept traffic from the load balancer machine. Consider the load balanced servers to be your proxy servers for this configuration.
   f. Click **Apply**, and then click **Save**.
7. Modify the SIP proxy transports from the administrative console.
   a. From the administrative console, click **Servers** > **Server Types** > **WebSphere proxy servers** > ***proxy_server_name*** > **SIP container transport chains > UDP_SIP_PROXY_CHAIN** > **UDP Inbound Channel**.
   b. Under the SIP proxy transports section, modify UDP_SIP_PROXY_CHAIN.
   c. From the Port menu, select SIP_LB_Address.
   d. Click **Apply**, and click **Save**.
8. Restart the proxy server to save your changes.

SIP proxy settings

The SIP proxy settings page contains general configuration items that affect outbound transport configuration, toleration of IP Sprayer devices, and access logging configuration.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> SIP proxy settings**.

The **SIP proxy settings** panel allows you to define attributes and policies that control the behavior of the SIP proxy server.

Default cluster

Specifies the default cluster name.

The **Default cluster** field must be set to a valid cluster before any SIP messages are routed through the proxy server. The default cluster indicates which cluster of application servers should receive SIP traffic when there are no cluster selection rules defined, or when none of the existing cluster selection rules match.

Data type	Valid cluster name
Default	None

Retry-after header value

Specifies the amount of time, in seconds, that the SIP proxy waits before it returns the SIP request because the SIP containers are overloaded, or the SIP proxy cannot locate available servers to which it can route the request.

Range	1 to 1000
Data type	Integer
Default	5

Logging

Specifies whether to enable or disable access logging.

When **Enable access logging** is checked, specifies the access log maximum size and proxy access log path name. When this control is unchecked, access log maximum size and proxy access log settings are disabled.

Data type	Boolean
Default	Unchecked

`Access log maximum size`
Specifies the maximum size, in megabytes (MB), of the access log before it rolls over.

Range	1 to 65535
Data type	Integer
Default	20

`Proxy access log`
Specifies the location of the SIP proxy access log.

Range	Valid path name
Data type	String
Default	`$(SERVER_LOG_ROOT)/sipproxy.log`

Container facing network interface

This section contains fields that configure the container-facing network interfaces.

Note: The value asterisk (*) means let the OS decide which interface to use.

`UDP interface`
Specifies the network interface for all User Datagram Protocol (UDP) data that goes to and from the SIP container. The value for this setting is the hostname or IP address to use for all communications between the SIP proxy and the SIP containers when the network is segmented.

Note: If a value is specified for this setting, you must also specify a value for the UDP port setting.

Data type	String
Default	*

`UDP port`
Specifies the UDP port for SIP container communications, such as the specific port required to pass through a firewall that separates the SIP proxy and the SIP containers.

Note: If a value is specified for this setting, you must also specify a value for the UDP interface.

Range	1 to 65535

Data type	Integer
Default	*

`TCP interface`
Specifies the network interface for all Transmission Control Protocol (TCP) data that goes to and from the SIP container or containers.

Data type	String
Default	*

`TLS interface`
Specifies the network interface for all Transport Layer Security (TLS) data that goes to and from the SIP container or containers.

Data type	String
Default	*

Load balancer health checking

This section contains fields that configure the load balancer used to load balance the SIP proxy server and perform health checks.

`Load balancer members (IP address 1 and 2)`
Specifies the IP addresses of the load balancers that source the SIP health checks. The IP addresses (1 and 2) ensure that the message is an actual SIP advisor request before sending a response back to it.

Data type	String
Default	None

`SIP health check method name`
Specifies the health check method name sent to the SIP proxy from the load balancer.

Data type	String
Default	None

`Tolerate a specific number of negative health checks`
Specifies whether to allow negative health checks.

Data type	Boolean
Default	False

`Maximum negative health checks`
Specifies the number of SIP health checks that the load balancer can perform that return negative results, indicating that there is a communication problem between the load balancer and the SIP proxy, before this SIP proxy informs the other SIP proxies that it is no longer processing data.

Range	1 through 10
Data type	Integer
Default	3

SIP network outage detection

Specifies whether to enable or disable outage detection.

When **Enable SIP network outage detection** is checked, specifies the KEEPALIVE interval and maximum KEEPALIVE failures. When this control is unchecked, network outage detection is disabled.

Data type	Boolean
Default	Unchecked

`Keep alive interval`

Specifies the interval, in milliseconds, at which the KEEPALIVE packets are sent to the SIP containers.

Range	1 through 1000
Data type	Integer
Default	3000

`Maximum keep alive failures`

Specifies the number of KEEPALIVE requests that are sent without getting a response before considering this server down.

Range	1 through 10
Data type	Integer
Default	3

Overload protection

Specifies whether to enable or disable overload protection.

When **Enable proxy managed overload protection** is checked, specifies the maximum throughput allowed and overload error response. When this control is unchecked, maximum throughput and overload error response fields are disabled.

Data type	Boolean
Default	False

`Max throughput factor`

Specifies the degree of overload protection at the proxy and is a percentage of the maximum messages per averaging period for the SIP container.

Range	1 through 100
Data type	Integer
Default	90

`Overload response code`

Specifies the overload response code. When the proxy detects an overload condition, it will respond to the request with the overload response code, if it is set.

Data type	Integer
Default	503

`Overload response reason`

Specifies the overload response reason phrase. When the proxy detects an overload condition, it will respond to the request with the overload response phrase, if it is set.

Data type	String
Default	Service unavailable

SIP external domains collection

The external domain collection panel provides create, remove and update capabilities for the external domain routing configuration.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> SIP proxy settings > External Domains**.

New
Create a new external domain entry. Clicking **New** launches the External Domain Detail panel.

Delete
Used to remove an external domain entry.

Domain
The domain string that is mapped to the associated protocol, host, and port configuration.

Distinguished name
The name that is associated with the external domain. It is used when SSL client authentication is enabled to limit connections from an external domain.

Protocol
This field contains the host name that the SIP Proxy will write to outbound requests so that the receiving agent will connect back to the IP Sprayer.

Host
The host used to make the SIP connection associated with the domain.

Port
The port used to make the SIP connection associated with the domain.

SIP external domains
The external domain detail panel configures the properties for external domain routing.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> SIP proxy settings > External Domains > New**.

Domain
The SIP domain that is mapped to the protocol, host, and port that is specified in the fields on this panel. The SIP proxy server matches the domain that is found in the TO header of a SIP message to this value and uses the related information to connect to the specified SIP service.

Range	Valid SIP domain name, with the addition of an optional preceding * as a wildcard.
Setting recommendations	None

Protocol
The protocol that the SIP proxy server uses to connect to the SIP service.

Range	TCP, SSL, and UDP
Default	TCP
Setting recommendations	None

Distinguished name
The name that is associated with the external domain. Used when SSL client authentication is enabled to limit connections from an external domain.

Range	Any string
Default	blank
Setting recommendations	None

Host

The host that the SIP proxy server uses to connect to the SIP service.

Range	Valid host name or IP address
Default	blank
Setting recommendations	None

Port

The port that the SIP proxy server uses to connect to the SIP service.

Range	1 to 65535
Default	blank
Setting recommendations	None

SIP routing rules collection

Routing rules enable an administrator to direct Session Initiation Protocol (SIP) traffic to a specific cluster when there is more than one cluster running SIP applications in a WebSphere Application Server Network Deployment cell.

Routing rules are not needed in cases where there is one cluster running SIP. Rules are only applied to the initial message of a SIP conversation and not all SIP traffic. To view this administrative console page, click **Servers** > **Server Types** > **WebSphere proxy servers** > *proxy_server_name* > **SIP proxy settings** > **Routing rules**.

New

Clicking **New** launches the Routing Rule Detail Panel for creating a new rule.

Delete

Deletes selected rules.

Enable

Sets the enabled attribute of the selected rules to true.

Disable

Sets the enabled attribute of the selected rules to false.

Set order

Launches the Set Order panel.

Select

Allows the user to select multiple rows to be affected by the Delete, Enable and Disable actions.

Order

The order column represents the order in which the rules are evaluated. This is critical since a SIP message may match more than one rule.

Cluster

The name of the cluster to which the rule will route SIP traffic.

Condition

A concatenation of the list of conditions associated with the rule. Condition is not sorted.

Enabled/Disabled

Indicates whether the rule is enabled, and thus considered for evaluation.

SIP routing rules set order

It is possible that a SIP message can match more than one routing rule but the SIP proxy will stop evaluating at the first match. Routing rules are evaluated in the order that they appear in the configuration file. The set order routing rule panel enables the administrator to change this ordering.

Routing rules are not needed in cases where there is one cluster running SIP. Rules are only applied to the initial message of a SIP conversation and not all SIP traffic. To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> SIP proxy settings > Routing rules > Set Order**.

Move Up

Clicking **Move Up** moves the selected rule up one row.

Move Down

Clicking **Move Down** moves the selected rule down one row.

Select

Enables the user to select one row that will be acted on by the Move Up and Move Down actions.

Cluster

The name of the cluster that the rule will direct SIP requests to.

Condition

A concatenation of the list of conditions that are associated with the rule.

Enabled

Indicates whether the rule is enabled and thus considered for evaluation.

SIP routing rules detail

The routing rule detail panel provides the ability to create and modify individual rules. Since the structure of conditions is complex, the user will need to utilize a different set of panels to change the conditions associated with a rule.

Routing rules are not needed in cases where there is one cluster running SIP. Rules are only applied to the initial message of a SIP conversation and not all SIP traffic. To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> SIP proxy settings > Routing rules > New**.

Enabled

Enables a rule to be removed from consideration without requiring that it be deleted. This field is checked by default.

Data type	Boolean
Default	true
Setting recommendations	none
Setting dependencies	none

Target Cluster

The name of the cluster that the SIP message will be routed to if the message attributes match the conditions.

Data type	String
Default	First cluster in the list.
Range	List of existing clusters or “null cluster” to indicate that a request should be rejected.

Setting recommendations	None
Setting dependencies	None

SIP rule condition collection

Each rule contains a list of conditions that are combined using a logical AND operator. This means that all of the conditions need to be true for the rule to apply. This panel enables the user to manage the set of conditions.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> SIP Proxy Server Settings > Routing rules >** *target_cluster* **> Conditions**.

New

Create a new condition. Clicking **New** launches the Rule Condition Detail panel.

Delete

Removes a condition from the collection.

Type

Indicates which aspect of a SIP message the condition applies to.

Value

The value that will be compared to the aspect of the SIP message indicated by type.

SIP rule condition detail

Routing rules direct SIP requests that match the condition to the selected cluster. The condition setting specifies which SIP messages match the rule. Rules only apply to the initial message of a SIP conversation and not all SIP traffic. Use this panel to create or modify a rule.

Note: Routing rules are not needed in cases where there is only one cluster running SIP.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> SIP Proxy Server Settings > Routing rules >** *target_cluster* **> Conditions > New**.

Condition type: Method

Select this option to use **Condition type: Method**. Selecting this condition type activates the Condition value field.

Default	Selected

Condition value

Specifies a fixed list of predefined method types when **Condition type: Method** is selected.

Range	INVITE, REGISTER, REFER, SUBSCRIBE, UNSUBSCRIBE, PUBLISH, MESSAGE, OPTIONS, INFO
Default	INVITE

Condition type: Defaults

Select this option to use **Condition type: Defaults**. Selecting this condition type activates the Type and Condition value fields. You can select between the fixed condition list, **Type**, and free form entry, **Condition value**, for this condition type.

Default	Not selected

Type
Specifies a fixed list of predefined attributes when **Condition type: Defaults** is selected.

Range	TO, FROM, REQUEST URI, SOURCE ADDRESS, DESTINATION ADDRESS
Default	TO

Condition value
Specifies a generic SIP message condition when **Condition type: Defaults** is selected.

Default	Empty string

Condition type: Header

Specifies whether to enable the header for a condition rule. You must specify a header name and header value pair if this setting is enabled.

Default	Not selected

Header name
Specifies the header name for the condition rule. If the value specified for this setting is *Header1*, then a packet sent to the proxy with this name specified as the header name satisfies the condition for the defined rule.

Default	Empty string

Header value
Specifies the header value for the condition rule. If the value specified for this setting is *Value1*, then a packet sent to the proxy with this value specified as the header value satisfies the condition for the defined rule.

Default	Empty string

SIP proxy inbound channel detail

This panel displays the configuration details of the SIP proxy inbound channel.

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere proxy servers** > *proxy_server_name* > **Transport chain> SIP proxy inbound channel** .

Transport channel name

Specifies the name of the SIP proxy inbound channel.

The name field cannot contain the following characters: `# \ / , : ; " * ? < > | = + & % '`

This name must be unique across all channels in a WebSphere Application Server environment. For example, a SIP proxy inbound channel and a TCP transport channel cannot have the same name if they reside within the same system.

Data type	String

Discrimination weight

Specifies the priority this channel has in relation to the other channels in this transport chain. This property is only used when port sharing is enabled and the channel chain includes multiple channels to which it might forward data. The channel in the chain with the lowest discrimination weight is the first one given the opportunity to look at incoming data and determine whether or not it owns that data.

Data type	Positive integer
Default	0

Troubleshooting the proxy server

This topic helps you to solve problems that you might encounter with your proxy server.

About this task

Proxy server errors are logged in the SystemOut.log, proxy.log, or local.log files. Consult the following list if you are having problems with your proxy server.

Note: This topic references one or more of the application server log files. Beginning in WebSphere Application Server Version 8.0 you can configure the server to use the High Performance Extensible Logging (HPEL) log and trace infrastructure instead of using `SystemOut.log` , `SystemErr.log`, `trace.log`, and `activity.log` files or native z/OS logging facilities. If you are using HPEL, you can access all of your log and trace information using the LogViewer command-line tool from your server profile bin directory. See the information about using HPEL to troubleshoot applications for more information on using HPEL.

Procedure

- **The proxy server was created successfully, but I am unable to start it.** Check the SYSOUT file for port conflicts. Use the **netstat –a** command to see if any of the endpoints that are associated with the proxy server are already being used. You can find the ports in the administrative console by clicking **Servers > Proxy servers >** *<server_name>* **> Ports**.

  If the proxy server fails to start when attempting to start it as a non-privleged user on UNIX systems, check for the following message in the logs:

  ```
  ChannelFramew E   CHFW0029E: Error initializing chain HTTPS_PROXY_CHAIN because of
  exception
  com.ibm.wsspi.channel.framework.exception.RetryableChannelException: Permission
  denied
  TCPPort       E   TCPC0003E: TCP Channel TCP_7 initialization failed.  The socket
  bind failed for host * and port 80.  The port may already be in use.
  ```

  Change the ports of the PROXY_HTTP_ADDRESS and PROXY_HTTPS_ADDRESS transport chains to values greater than `1024`.
- **The proxy server routes requests to the web container over an administration port.** The proxy server is located in front of several web containers. The configuration requires that the web containers listen to the non-default ports such as 9061, 9081, and so on. This scenario is the default case when multiple application servers are on the same machine, which forces new and different ports to be used in the configuration. In this scenario, the proxy server might route an application request to the web container over the administration port of 9061, instead of using the expected port of 9081.

  Add the listening port numbers of the web container to the virtual host that is associated with the target application. This process will ensure that the proxy server routes the request to the web container over the correct port number.
- **The proxy server started, but I am unable to access the application resources through the endpoints for the proxy server.** Ensure that the endpoints for the proxy server are among the host aliases in the virtual host that are associated with the application.
- **The proxy server routes to another core group.** Verify that core group bridges exist between the core groups in the cell, and that the processes that are chosen to be bridges are restarted. If there is a firewall between the core group, verify that the correct ports are open for core group bridge traffic.
- **The proxy server is unable to route requests to another cell.** Review the core group bridge settings. If the HMGR0149E error message is logged on any server, usually on a server acting as a core group bridge, then the security settings for the cell need to be modified to allow communication.

See the topic entitled *Exporting Lightweight Third Party Authentication keys* for more information about configuring cell security between multiple cells.

- **Receiving a blank page when making a request to the proxy.** Consider the following actions:
  - Update the virtual host. Ensure that the target application and routing rule are assigned to a virtual host that includes the proxy server listening ports (default: HTTP 80, HTTPS 443). Add the proxy server listening ports to the application, or routing rule virtual host, or use the proxy_host virtual host.
  - Stop the conflicting process. Check your system to ensure that no other process (for example, Apache, IBM HTTP Server, and so on) is running that uses the proxy server ports (default: HTTP 80, HTTPS 443). If this problem occurs, the proxy server seems to start normally, but is unable to receive requests on the affected listening port. Check your system as follows:
    1. Stop the proxy server.
    2. Query your system using **netstat** and **ps** commands to determine if an offending process is using the port on which the proxy server is listening.
    3. If an offending process is found, stop the process and configure your system so that the process is not started during system startup.
  - Enable proxy routing. Ensure that proxy routing is enabled for the web module of the application. Proxy routing is enabled by default, so if no proxy properties are modified, disregard this solution. Otherwise, see “Customizing routing to applications” on page 279 for instructions on modifying the proxy properties.
  - Test direct request. Ensure that the target application is installed by making a request directly to the application server. If a response is not received, then the problem is with the application server and not the proxy server. Verify this case by going through the proxy server after you can receive a response directly from the application server.
- **HTTP 404 (File not found) error received from the proxy server**. Consider the following actions:
  - Update the virtual host. Ensure that the target application and routing rule are assigned to a virtual host that includes the proxy server listening ports (default: HTTP 80, HTTPS 443). Add the proxy server listening ports to the application, or routing rule virtual host, or use the proxy_host virtual host.
  - Enable proxy routing. Ensure that proxy routing is enabled for the web module of the application. Proxy routing is enabled by default, so if no proxy properties are modified, disregard this solution. Otherwise, see “Customizing routing to applications” on page 279 for instructions on modifying the proxy properties.
  - Test direct request. Ensure that the target application is installed by making a request directly to the application server. If a response is not received, then the problem is with the application server and not the proxy server. Verify this case by going through the proxy server when you can receive a response directly from the application server.
- **Unable to make Secure Sockets Layer (SSL) requests to application or routing rule**. Ensure that the virtual host of the application or routing rule includes a host alias for the proxy server SSL port (default: 443).
- **Unable to connect to the proxy server...request times out**. Stop the conflicting process. Check your system to ensure that no other process (for example, Apache, IBM HTTP Server, and so on) is running that uses the proxy server ports (default: HTTP 80, HTTPS 443). If this situation occurs, the proxy server seems to start normally, but is unable to receive requests on the affected listening port. Check your system, as follows:
  1. Stop the proxy server.
  2. Query your system using **netstat** and **ps** commands to determine if an offending process is using a port on which the proxy server is listening.
  3. If an offending process is found, stop the process and configure your system so that the process is not started during system startup.
- **Did not receive a response from the error page application when the HTTP error occurred (for example, 404)**. Ensure that the error page URI is entered correctly. Also, make sure that the Handle remote errors option is selected if you are handling HTTP error responses from back-end servers. For

more detailed information, refer to “Overview of the custom error page policy” on page 287 and the custom error page policy section of “Proxy server settings” on page 228.

- **What packages do I enable when tracing the proxy server?** All of the following packages are not needed for every trace, but if unsure, use all of them:
  - *=info
  - WebSphere Proxy=all
  - GenericBNF=all
  - HAManager=all
  - HTTPChannel=all
  - TCPChannel=all
  - WLM*=all
  - DCS=all
  - ChannelFrameworkService=all
  - com.ibm.ws.dwlm.*=all
  - com.ibm.ws.odc.*=all
- **How do I enable SSL on/off load?** SSL on/off load is referred to as the transport protocol in the administrative console, and transport protocol is a web module property. Refer to “Customizing routing to applications” on page 279 to see how to configure web module properties. No SSL on/off load or transport protocol properties exist for routing rules because the transport protocol is inherent to the generic server cluster that is specified in the routing rule.
- **When fronted by IBM HTTP Server or a web server plug-in, how do I configure the proxy server so I do not have to add a port for it to the virtual host?** For the proxy server to trust the security-related information that is included in a request, such as private headers that the product adds, add the originator of the request to the proxy server trusted security proxies list. For example, add an IBM HTTP Server or a web server plug-in that sends requests to the proxy server, to the proxy server trusted security proxies list. The web server plug-in can then send product added private header information that, contains the virtual host information of a request. If the proxy does not trust these private headers from the web server plug-in, or from any client, the proxy server adds its own private headers, which requires the addition of proxy server HTTP and HTTPS ports to the virtual host. Typically, when a web server plug-in is used with the proxy server, the intent is to use the proxy server as a back-end server. Therefore, you must add the plug-in as a trusted security proxy to avoid having to expose the proxy server ports. “Routing requests from a plug-in to a proxy server” on page 284 provides more information about configuring a web server plug-in to use with the proxy server. “Proxy server settings” on page 228 provides more information about setting up trusted security proxies.
- **The proxy server seems to hang under stress, or Too Many Files Open exceptions display in ffdc or SystemErr.log.** Under high connection loads, the number of file system descriptors might become exhausted and the proxy server may seem to hang and drop Too Many Files Open exceptions in the `ffdc` directory or in the `SystemError.log` file because it is unable to open a socket. To alleviate this problem, modify one or more of the following parameters at the operating system level, and at the proxy server level to optimize the use of connections for the proxy server:
  - Windows **Operating system tuning for Windows 2003, and XP**
    - TcpTimedWaitDelay - Determines the time that must elapse before TCP/IP releases a closed connection and reuse its resources. This interval between closure and release is known as the TIME_WAIT state, or twice the maximum segment lifetime (2MSL) state. During this time, reopening the connection to the client and server costs less than establishing a new connection. By reducing the value of this entry, TCP/IP releases closed connections faster and can provide more resources for new connections. Adjust this parameter if the running application requires rapid release, the creation of new connections, or an adjustment because of a low throughput caused by multiple connections in the TIME_WAIT state.

      View or set this value as follows:

1. Use the **regedit** command and access the `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\TCPIP\Parameters` registry subkey to create a new REG_DWORD value named TcpTimedWaitDelay.
2. Set the value to decimal 30, which is Hex 0x0000001e. This value sets the wait time to 30 seconds.
3. Stop and restart your system.

Default value	0xF0, which sets the wait time to 240 seconds (4 minutes).
Recommended value	A minimum value of 0x1E, which sets the wait time to 30 seconds.

- MaxUserPort - Determines the highest port number that TCP/IP can assign when an application requests an available user port from the system. View or set this value as follows:
  1. Use the **regedit** command, access the `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\TCPIP\Parameters registry` subkey, and create a new REG_DWORD value named MaxUserPort.
  2. Set this value to at least decimal 32768.
  3. Stop and restart your system.

Default value	None
Recommended value	At least decimal 32768.

– Linux **Operating system tuning for Linux**

- timeout_timewait parameter - Determines the time that must elapse before TCP/IP releases a closed connection and can reuse its resources. This interval between closure and release is known as the TIME_WAIT state, or twice the maximum segment lifetime (2MSL) state. During this time, reopening the connection to the client and server costs less than establishing a new connection. By reducing the value of this entry, TCP/IP can release closed connections faster, providing more resources for new connections. Adjust this parameter if the running application requires rapid release, the creation of new connections, and a low throughput due to many connections sitting in the TIME_WAIT state.

  View or set this value by issuing the following command to set the timeout_timewait parameter to 30 seconds:

  ```
  echo 30 > /proc/sys/net/ipv4/tcp_fin_timeout
  ```
- Linux file descriptors (ulimit) - Specifies the number of open files that are supported. The default setting is typically sufficient for most applications. If the value set for this parameter is too low, a file open error, memory allocation failure, or connection establishment error might display.

  View or set this value by checking the UNIX reference pages on the ulimit command for the syntax of different shells. Set the ulimit command to 65535 for the KornShell shell (ksh), by issuing the ulimit -n 65535 command. Use the ulimit -a command to display the current values for all limitations on system resources.

Default value	1024
Recommended value	65535

– AIX **Operating system tuning for AIX**

- TCP_TIMEWAIT - Determines the time that must elapse before TCP/IP releases a closed connection and can reuse its resources. This interval between closure and release is known as the TIME_WAIT state, or twice the maximum segment lifetime (2MSL) state. During this time, reopening the connection to the client and server costs less than establishing a new connection. By reducing the value of this entry, TCP/IP can release closed connections faster, providing more resources for new connections. Adjust this parameter, if the running application requires rapid release or the creation of new connections, or if a low throughput occurs due to many connections sitting in the TIME_WAIT state.

View or set this value by issuing the following command to set the TCP_TIMEWAIT state to 15 seconds:

```
/usr/sbin/no -o tcp_timewait =1
```

- AIX file descriptors (ulimit) - Specifies the number of open files that are permitted. The default setting is typically sufficient for most applications. If the value set for this parameter is too low, errors can occur when opening files or establishing connections, and a memory allocation error might display. To prevent WebSphere Application Server from running short on resources, remove the upper limits (ulimit) for resources on the user account on which the WebSphere Application Server process runs.

  View or set this value by changing the ulimit settings as follows:

  1. Open the command window.
  2. Type **smitty users** to open the AIX configuration program.
  3. Select **Change** or **Show Characteristics** of a user.
  4. Type the name of the user account on which the WebSphere Application Server runs.
  5. Press **Enter**.
  6. Change the following settings to the indicated value:

Soft FILE Size	-1
Soft CPU Time	-1
Soft STACK Size	-1
Soft CORE File Size	-1
Hard FILE Size	-1
Hard CPU Time	-1
Hard STACK Size	-1
Hard CORE File Size	-1

  7. Press **Enter** to save changes.
  8. Log out and log into your account.
  9. Restart the product.

Default value	2000
Recommended value	unlimited

– HP-UX **Operating system tuning for HP-UX**

HP-UX nfile kernel parameter - Specifies the maximum number of files that can be open on the system at any given time. Not specifying a large enough number might restrict your system processing capacity.

HP-UX ninode kernel parameter - Specifies the maximum number of open inodes that can be in memory. An open inode is associated with each unique open file. Therefore, the value you specify for the ninode parameter should be larger than the number of unique files that must be open at the same time.

– Solaris **Operating system tuning for Solaris**

- Solaris file descriptors (ulimit) - Specifies the number of open files that are supported. If the value specified for this parameter is too low, a file open error, memory allocation failure, or connection establishment error might display. View or set this value by checking the UNIX reference pages on the ulimit command for the syntax of different shells.

Issue the `ulimit -a` command to display the current values for all limitations on system resources.

Issue the `ulimit -n 65535` command to set the ulimit command to 65535 for the KornShell shell (ksh).

The maximum number of files that can be open for a process is also affected by the `rlim_fd_max`, and `rlim_fd_cur` settings for the global kernel limit. You might need to increase the values of these settings in the /etc/system file.

Solaris 10 provides a new mechanism for setting kernel parameters. Issue one of the following commands to specify a new limit for the max number of file descriptor kernel parameter on Solaris 10:

1. To change the value for the current shell session:

   ```
   prctl -n process.max-file-descriptor -r -v 65535 $$
   ```

2. To make the change a system wide change:

   ```
   projmod -sK 'process.max-file-descriptor=(privileged,65535,deny)'
   system
   ```

   Caution should be used when issuing this command because this command changes the setting for all users, and all projects.

3. To change the value for all projects that are owned by user root:

   ```
   projmod -sK 'process.max-file-descriptor=(privileged,65535,deny)'
   user.root
   ```

   4. To change the value for all projects owned by non-root user

   ```
   projmod -sK 'process.max-file-descriptor=(privileged,65535,deny)'
   user.username
   ```

- TCP_TIME_WAIT_INTERVAL - Notifies TCP/IP on how long to keep the connection control blocks closed. After the applications complete the TCP/IP connection, the control blocks are kept for the specified time. When high connection rates occur, a large backlog of the TCP/IP connections accumulate and can slow server performance. The server can stall during certain peak periods. If the server stalls, the **netstat** command shows that many of the sockets that are opened to the HTTP server are in the CLOSE_WAIT or FIN_WAIT_2 state. Visible delays can occur for up to four minutes, during which time the server does not send any responses, but CPU utilization stays high with all of the activities in system processes.

  View or set this value by using the **get** command to determine the current interval and the set command to specify an interval of 30 seconds. For example:

  ```
  ndd -get /dev/tcp tcp_time_wait_interval
  ndd -set /dev/tcp tcp_time_wait_interval 30000
  ```

Default value	240000 milliseconds, which is equal to 4 minutes.
Recommended value	60000 milliseconds

– **Proxy server tuning**

- Persistent requests - A persistent request is one that is sent over an existing TCP connection. You can maximize performance by increasing the number of requests that are received over a TCP connection from a client. The value should represent the maximum number of embedded objects, for instance GIF and so on, in a web page +1.

  View or set this value in the WebSphere Application Server administrative console by clicking **Servers > Proxy Servers >** *server_name* **> Proxy server transports > HTTP_PROXY_CHAIN/ HTTPS_PROXY_CHAIN**

Default value	100
Recommended value	A value that represents the maximum number of embedded objects in a web page + 1.

- Outbound connection pool size - The proxy server pools outbound connections to target servers and the number of connections that resides in the pool is configurable. If the connection pool is depleted or empty, the proxy server creates a new connection to the target server. Under high concurrent loads, increase the connection pool size should to a value of the expected concurrent client load to achieve optimal performance.

  View or set this value in the WebSphere Application Server administrative console by clicking **Servers** > **Proxy Servers** > *server_name* > **HTTP Proxy Server Settings**. In the Content Server Connection section, increase the maximum connections per server field to a value that is equal to or greater than the expected maximum number of connected clients. Save your changes, synchronize the changes to the proxy server node, and restart the proxy server.

Recommended value	Value consistent to the expected concurrent client load.

- Outbound request time-out - Often times, the back-end application servers that are fronted by the proxy server may be under high load and may not respond in an adequate amount of time, therefore the connections on the proxy server may be tied up from waiting for the back-end application server to respond. Alleviate this by configuring the amount of time the proxy server waits for a response from the target server. This is the Outbound Request Time-out value. By managing the amount of time the proxy server waits for a slow back-end application server, connections are freed up faster and used for other request work.

  View or set this value in the administrative console by clicking **Servers** > **Server Types** > **WebSphere proxy servers** > *proxy_server_name* > **HTTP Proxy Server Settings**. In the Content Server Connection section, set Outbound Request Time-out to a value that represents the acceptable response time from the point of view of the client.

Default value	120
Recommended value	A value that represents the acceptable response time from the point of view of the client.

Troubleshooting request routing and workload management through the proxy server

This section provides information for how to troubleshoot request traffic that flows through the proxy server.

Before you begin

You will need to know the machines and nodes that will belong to the proxy server cluster, because the product needs to be installed on those machines. You will also need to know the URL for the applications, application deployment, and cluster definition details. The proxy server should be started.

About this task

You can use the proxy server MBean to determine how requests are routed to applications, and subsequently, to a particular application server. If the request is being routed incorrectly, you can disable routing to specific applications or reconfigure the routing rules.

Note: This topic references one or more of the application server log files. Beginning in WebSphere Application Server Version 8.0 you can configure the server to use the High Performance Extensible Logging (HPEL) log and trace infrastructure instead of using `SystemOut.log` , `SystemErr.log`, `trace.log`, and `activity.log` files or native z/OS logging facilities. If you are using HPEL, you can access all of your log and trace information using the LogViewer command-line tool from your server profile bin directory. See the information about using HPEL to troubleshoot applications for more information on using HPEL.

Procedure

1. Obtain the Dynamic Route MBean for the proxy server and invoke the operation to generate routing information for the URI. Start **wsadmin** and get all of the Dynamic Route MBeans as follows:

   ```
    $AdminControl queryNames
    type=DynamicRoute,*

   set routembean <cut and paste the MBean Identifier from the previous command output>

    $AdminControl invoke $routembean debugRouting {http://*/urlpattern all}
   ```

   Use an asterisk (*) to match all of the virtual hosts, or explicitly specify a virtual host. For example, `http://proxy_name:80/urlpattern`. The **set routembean** command should correspond to the MBean from the output of the previous command.

   The proxy server will start generating routing-related information for all subsequent HTTP requests that match the specified virtual host and URL pattern to the `SystemOut.log` file.
2. Send representative workload traffic through the proxy server.
3. Analyze the routing information in the proxy server `SystemOut.log` file.
4. Make required changes to application routing to enable or disable routing through the proxy server, using the administrative console, by clicking **Applications > Enteprise Applications**.
5. Repeat steps two through four until the routing of all requests are satisfied.
6. Disable gathering routing information using **wsadmin** as follows:

   ```
   $AdminControl invoke $routembean
   stopDebugRouting
   ```

Results

The proxy server and the applications are correctly configured for external access.

Session Initiation Protocol overload protection

Session Initiation Protocol (SIP) overload protects the system from two overload conditions: memory overload and CPU overload. Container managed overload protection (CMOP) and proxy managed overload protection (PMOP) allow for real-time protection based on the overload settings information.

SIP container managed overload protection

In a stand-alone server deployment, SIP container managed overload protection (CMOP) provides the only defense against both memory and CPU overload conditions. An administrator can set up several static thresholds using SIP container settings. When these thresholds are exceeded, the container begins to drop new requests by responding to any requests that initiate new dialogs with a 503 response until the container is no longer overloaded. This includes settings that affect memory and CPU usage.

In an ND deployment, CMOP allows the container to use the same SIP container settings to notify the proxy server when the container is in an overloaded state. After the proxy server receives this notification, it begins to drop new requests instead of forwarding them to the containers. All memory overload conditions in ND are prevented by CMOP regardless of the configuration.

Overload protection is calculated based on the virtual memory settings and the maximum throughput that a container can handle. You can specify a value for the following SIP container settings from the administrative console for CMOP.

- Maximum application sessions
- Maximum messages per averaging period
- Maximum response time
- Maximum dispatch queue size

You can also specify a value for these SIP container custom properties for CMOP.
- thread.message.queue.max.size
- weight.overload.watermark

SIP proxy managed overload protection

SIP Proxy Managed overload protection (PMOP) is the best line of defense against container overload. In a typical deployment, CMOP alone does not provide optimal results. The following conditions might occur if CMOP is deployed without deploying PMOP at the proxy server.
- The on or off mechanism might become too granular
- Admission rates might fluctuate
- An absolute cap on load might be difficult to establish
- Unstable loads might be sent to the containers

When PMOP is deployed, the proxy server utilizes admission rate controllers for each container. When a container is overloaded, instead of either accepting or rejecting new load for a period of time, new workload can be sent to the backend containers without completely shutting down the flow of new traffic. This allows the proxy server to offer a consistent load to the containers without exceeding the maximum values for the container settings.

The SIP proxy server calculates a maximum value for message throughput to each backend container based on a percentage of the configured Maximum Messages per Averaging Period (MMAP) setting specified for the container. The maximum value for message throughput is called the Maximum Throughput Factor (MTF).

MTF is disabled by default and can only be enabled by specifying a value for the maxThroughputFactor custom property. The value specified for the MTF custom property should be less than 100 percent to prevent CPU overload at the container. For example, you might set this value to 90 percent.

When the value of the MTF custom property is set to less than 100 percent, the total throughput to the container should never exceed the maximum value specified for the MMAP container setting. This process protects the container from handling excessive loads when coming out of an overloaded condition.

The MTF value should always be specified when stable and accurate overload protection is required. Specifying the MTF setting provides the best results for loads that range up to twice the capacity of the system. You should consider your system capacity when configuring for overload protection.

The algorithm utilized by the SIP proxy server relies on several features to provide stable and accurate loads to the backend container.
- Per-server rate control managed at the proxy server
- Auto-adjusting, per second admission rate controller
  - Ratio of in-dialog to non-dialog averages used to control rate
  - Auto-rate reduction when in an overloaded state
- Ability to absorb fast load transitions
- Burst tolerance to allow for short occasional load burst without triggering overload
- Stabilization control to prevent excessive overload when there is a transition in the cluster

You can specify a value for the following SIP proxy server custom properties for PMOP.
- burstResetFactor
- deflatorRatio
- dropOverloadPackets
- inDialogAveragingPeriod

- maxThroughputFactor
- outDialogAveragingPeriod
- perSecondBurstFactor
- proxyTransitionPeriod
- sipProxyStartupDelay

Configuring SIP quorum support using the default core group

You can configure quorum to avoid inconsistency among Session Initiation Protocol (SIP) containers, or repeated errors from a proxy server. If quorum is enabled before a network partition occurs, the correct routing decisions can be made. A network partition can occur when a set of SIP containers are disconnected from the network and then reconnected.

Before you begin

Determine whether you want to configure the quorum feature for an entire core group, or for specific clusters within a core group. The topic *Implications of high availability group policy settings* provides additional information about quorum functionality.

About this task

All SIP containers publish a set of unique identifiers (IDs) to the SIP proxy server. Each ID represents a set of SIP sessions. The SIP proxy server is able to make the correct routing decisions based on the one-to-one mapping of these IDs to the SIP containers.

A network partition occurs whenever a network device within the topology of the cell fails. As a result, some portion, or partition, of the cell disconnects from the other portion, or partition.

gotcha: If a network partition evenly splits the cluster members, half of the cluster members are arbitrarily selected to be in the quorum. This split might cause a restart of the partition while it is still connected to clients. Therefore, you should configure your system to minimize evenly split partitions. You should create a set of three groupings: three data centers, three blade centers, and three groupings of cluster members.

Whenever network connectivity is interrupted, a backup SIP container takes ownership of all IDs that were managed by the disconnected SIP container. The backup container then publishes ownership of these IDs to the SIP proxy server so that the proxy server can make the correct routing decisions.

When the network connection for the primary container is restored, the primary container begins publishing ID ownership to the SIP proxy server to indicate that it owns the same IDs as the backup container. If the SIP proxy server has two destinations for each ID, it is impossible for the SIP proxy server to consistently make the correct routing decisions. To avoid this problem, you must configure the SIP quorum feature for your proxy servers before a network partition occurs.

The SIP quorum feature can be enabled for an entire core group, such as DefaultCoreGroup, or for specific clusters within a core group. When the quorum feature is enabled on the Default SIP Quorum core group policy for a core group, the feature is enabled for all clusters in the core group. If the SIP quorum feature is only enabled for some of the clusters in a core group, then you must modify the default policy, and create a duplicate core group policy setting to enable quorum support for those clusters.

You must enable the Default SIP Quorum core group policy, or configure a duplicate core group policy with quorum enabled for each core group that requires quorum support. The purpose of the high availability group that uses the core group policy is to track quorum between the SIP containers, or SIP proxy servers, in a cluster.

Complete these steps to configure the quorum feature using an additional SIP quorum core group policy.

Note: Only one cluster name can be applied to a policy. Repeat this procedure for each cluster that requires the SIP quorum feature to create a new policy with the appropriate match criteria. Each SIP quorum policy should have at the most three match criteria.

Procedure

1. In the administrative console, click **Servers > Core Groups > Core group settings >** *core_group_name*.
2. Click **Policies > New**.
3. Select **All active policy**, and then click **Next**
4. Specify a unique name for the policy in the **Name** field, and then enter a description in the **Description** field.
5. Select **Quorum**, and then click **OK**. You will receive a warning message that indicates that you must define at least one match criteria for this policy.
6. In the Additional Properties section, click **Match criteria > New**.
7. Specify `policy` in the **Name** field, and `AllActiveQuorumPolicy` in the **Value** field. Both of these fields are case sensitive, and both values must be entered as shown in this step.
8. Click **Apply**.
9. Click the name of your policy in the navigation breadcrumb on this administrative console page, and then, in the Additional Properties section, click **Match criteria > New** again.
10. This time, specify `type` in the **Name** field, and `SIP_QUORUM` in the **Value** field. As previously mentioned, both of these fields are case sensitive, and both values must be entered as shown in this step.
11. Click **Apply**.
12. Perform the following actions if you only want this new policy to apply to a specific cluster within the core group. Skip this step if you want this new policy to apply for the entire core group.
    a. Click the name of your policy in the navigation breadcrumb on this administrative console page, and then, in the Additional Properties section, click **Match criteria > New** again.
    b. This time, specify `IBM_hc` in the **Name** field, and the name of a cluster, to which you want to apply this policy, in the **Value** field. Both of these fields are case sensitive. Therefore, make sure that you enter the cluster name exactly as it is defined in the core group.
    c. Click **Apply**.
13. Click **Save** to save your configuration changes.
14. Restart the affected servers.

    If the new SIP quorum core group policy applies to the entire core group, you must restart every server that is part of that core group.

    If the new SIP quorum core group policy only applies to a specific cluster within the core group, you must restart all of the members of that cluster.
15. Repeat these steps if you are only enabling SIP quorum for specific clusters in the core group.

    Only one cluster name can be associated with a policy. If you have multiple clusters in this core group, for which you want to enable SIP quorum, but you are not enabling SIP quorum for the entire core group, you must create an entirely new policy for each cluster for which you are enabling SIP quorum. For each new policy, you must specify the three match criterias `type=SIP_QUORUM`, `policy=AllActiveQuorumPolicy`, and `IBM_hc=`*cluster_name*.

Results

The majority of the members that are included in a SIP container or a proxy server cluster must be started before quorum is achieved. The members of a proxy server cluster do not start to listen for SIP requests until quorum is achieved.

When a SIP proxy server is disconnected from the network, a SIP proxy server in the minority partition of the cell continues to handle SIP requests. The minority partition of a cell is the partition that has connectivity to the fewest cluster members.

When a SIP container is disconnected from the network, the SIP container in the minority partition is automatically restarted to clear all information about the logical partitions that were previously managed. The SIP containers in the majority partition take ownership of the logical partitions from the disconnected SIP container and publish these IDs to the SIP proxy servers.

Configuring the SIP proxy for network outage detection

When SIP traffic and the high availability (HA) manager traffic are not on the same network, failures that affect SIP traffic might not be detected by the HA manager, resulting in loss of SIP traffic. You can configure network outage detection on the same network as the SIP traffic to ensure that failures that affect SIP traffic can be detected by the HA manager.

About this task

Define these settings for network outage detection to ensure that SIP network failures are detected by the HA manager.

Procedure

1. From the administrative console, select **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> SIP proxy settings**. Scroll to the SIP network outage detection settings on this page.
2. Select **Enable SIP network outage detection** and enter the **Keep alive** interval. The proxy will send keep alive messages to the server at the configured keep alive interval.
3. Specify the **Maximum keep alive failures** to send before a server failure response is sent.
4. Optional: Set the numFailuresSipAdvisorRequests custom property so that the SIP proxy notices the failure to receive load balancer advisor requests.

Results

If the proxy misses the number of responses configured as keepAliveFailures, or the server misses that number of keep alive requests, it will determine that SIP traffic should no longer be sent between the proxy and the server. The proxy will stop sending SIP messages to that server, and the server will restart and wait for connectivity to be restored before resuming operation.

Administering proxy actions

You can create new proxy actions or you can administer an existing proxy action. A proxy server action is an event that is performed when an HTTP request or an HTTP response is received by the a proxy virtual host. Some examples of proxy server actions include caching actions, rewriting actions, compression actions, header modification actions, and routing actions.

Before you begin

A proxy action cannot be performed unless it is associated with a proxy rule expression. A proxy rule expression is only evaluated when it is associated with a proxy virtual host. A proxy action can be created or administered without a proxy rule expression or a proxy virtual host, but it cannot be used without them.

About this task

Proxy actions are associated with proxy rule expressions. If a proxy rule expression evaluates to true, then all of the proxy actions specified in the proxy rule expression configuration are performed. Complete these steps to create a new proxy action or to administer an existing proxy action for the proxy server.

Procedure

1. From the administrative console, select **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> Proxy actions**.
2. Optional: Manage an existing proxy action configuration.
   a. Click *proxy_action_name* if you want to view or modify the settings for an existing proxy action. If you have made changes to the proxy action configuration, click **OK** to save the changes.
   b. Click **Delete** to remove an existing proxy action that has been selected.
3. If you want to create a caching proxy action, then click **New Caching Action**.
   a. Enter the name of the proxy action in the Action name field. The Action name is used as the unique identifier for this proxy action configuration. The Action name must be unique with the cell and cannot include any of the following characters: # \ / , : ; " * ? < > | = + & % '.
   b. Select the **Enable caching** check box to enable caching.
   c. Enter the value in seconds for the **Default expiration** field. The Default expiration field specifies the amount of time, in seconds, before a cached response expires.
   d. Enter the value in seconds for the **Last modified** factor. The Last modified factor field specifies the amount of time before a response is cached if the response does not have explicit HTTP expiration headers.
   e. Click **OK** to save the proxy action configuration.
4. If you want to create a compression action, then click **New HTTP Request Compression Action** to create an action for requests, or click **New HTTP Response Compression Action** to create an action for responses.
   a. Enter the name of the proxy action in the **Action name** field. The Action name is used as the unique identifier for this proxy action configuration. The Action name must be unique with the cell and cannot include any of the following characters: # \ / , : ; " * ? < > | = + & % '.
   b. Select the appropriate compression type from the Compression type menu.
   c. Select the content type for which compression should occur. You can select multiple content types to be compressed. Click **New** to add a new content type to the list. If you want to remove a content type from the list, then select the content type to be removed, then click **Delete**.
   d. Click **OK** to save the proxy action configuration.
5. If you want to create a header action, click **New HTTP Request Header Action** to create an action for requests, or click **New HTTP Response Header Action** to create an action for responses.
   a. Enter the name of the proxy action in the **Action name** field. The Action name is used as the unique identifier for this proxy action configuration. The Action name must be unique with the cell and cannot include any of the following characters: # \ / , : ; " * ? < > | = + & % '.
   b. Enter the name of the HTTP header to be modified in the **Header name** field.
   c. Select the appropriate action from the **Header modify action** menu. The available header modify actions include:
      - Set
      - Append
      - Edit
      - Remove
   d. Enter the value to be used for the Header modify action in the **Header value** field.
   e. Enter an expression to be performed on the header value in the **Header value expression** field. The Header value expression is evaluated and the modify action is performed if the evaluation returned a match.

f. If you are creating a HTTP Request Header action, then select the appropriate methods for the Header modify action being performed. You can select multiple methods. Click **New** to add a new method to the list. If you want to remove a method from the list, then select the method to be removed and click **Delete**.

g. If you are creating a HTTP Response Header action, then select the appropriate status codes for the Header modify action being performed. You can selected multiple status codes to be included. Click **New** to add a new status code to the list. If you want to remove a status code from the list, select the status code to be removed and click **Delete**.

h. Click **OK** to save the proxy action configuration.

6. If you want to create a rewriting action proxy action, then click **New Rewriting Action**.

   a. Enter the name of the proxy action in the **Action name** field. The Action name is used as the unique identifier for this proxy action configuration. The Action name must be unique with the cell and cannot include any of the following characters: # \ / , : ; " * ? < > | = + & % '.

   b. Select the type of rewriting action to be performed in the Rewriting action types menu. The follow elements can be rewritten using this type of proxy action:
      - Absolute URL response
      - Redirect location header
      - Redirect status code
      - Relative URL response
      - Set cookie domain
      - Set cookie path

   c. Enter the subject URL pattern to be rewritten in the **From pattern** field.

   d. Enter the resulting URL pattern after the rewrite has occurred in the **To pattern** field.

   e. Optional: Select **Enable passive rewrite** to defer the rewriting of the URI until the subsequent request for that URI is sent by the client.

   f. Optional: In the **Cookie name** field, enter the name of the cookie for which the domain or the path are to be rewritten. This field is only valid for the Set-Cookie type of rewriting actions.

   g. Optional: In the **Limit URL pattern** field, specify a constraint on the URL patterns to rewrite in the response message. Limiting the URL pattern prevents the proxy server from rewriting all URL patterns in the response message of a certain page. This field is only valid for the absolute URL response action type or the relative URL response action type.

   h. Optional: In the **Limit Cookie domain** field, specify a constraint to limit the rewriting of the cookie domain to only a set of specified domains. If no domains are specified, then all domains are rewritten. This field is only valid for the Set-Cookie type of rewriting action.

   i. Optional: In the **Limit Cookie path** field, specify a constraint to limit the rewriting of the cookie path to only a set of specified paths. If no paths are specified, then all the paths are rewritten. This field is only valid for the Set-Cookie type of rewriting action.

   j. Click **OK** to save the proxy action configuration.

7. If you want to create a routing proxy action, then click one of the following: **New Application Server Route**, **New Generic Server Cluster Route**, **New Fail Route**, **New Redirect Route**, or **New Local Route**.

   a. Enter the name of the proxy action in the **Action name** field. The Action name is used as the unique identifier for this proxy action configuration. The Action name must be unique with the cell and cannot include any of the following characters: # \ / , : ; " * ? < > | = + & % '.

   b. If you are creating a new application server route, then follow these additional steps:

      1) Enter the beginning time for this routing rule in the **Start time** field. If the start time is specified using a 12-hour clock, then click **AM** or **PM**. If the start time is specified using a 24-hour clock, then click **24-hour**.

2) Enter the finishing time for this routing rule in the **End time** field. If the end time is specified using a 12-hour clock, then click **AM** or **PM**. If the end time is specified using a 24-hour clock, then click **24-hour**.
3) Select **Include** or **Exclude** from the Action menu to specify the type of rule being configured.
4) Select the applications servers that will follow this rule from the Available application servers menu.
5) Click **>**.
6) If you want to remove an application server from the Enabled application servers menu, then select that server from the Enabled application servers menu and click **<**.

c. If you are creating a new generic server cluster route, then follow these additional steps:
   1) Select the Generic server cluster that will follow this rule from the **Generic server cluster name** menu.
   2) Select either **Active Affinity** or **Passive Affinity** for the affinity type.
   3) If you selected Active Affinity, then in the **Default expiration** field, enter the expiration time in seconds.
   4) If you selected Passive Affinity, then in the **Cookie name** field, enter the name of the cookie to be used by the proxy server to manage affinity.
   5) If you selected Passive Affinity, then select the generic server mappings for which this cookie will be used to manage affinity. If you need to create any additional mappings, click **New**. If you need to remove existing cookie mappings, select the appropriate mapping and click **Delete.**
   6) Click **New Time Mapping**.
   7) Enter the beginning time for this routing rule in the **Start time** field. If this rule should be applied all the time, then selected **24-hour**.
   8) Enter the finishing time for this routing rule in the **End time** field. If this rule should be applied all the time, then selected **24-hour**.
   9) Select **Include** or **Exclude** from the Action menu to specify the type of rule being configured.
   10) Select the cluster members that will follow this rule from the Available generic server cluster members menu.
   11) Click **>**.
   12) If you want to remove cluster member from the Enabled generic server cluster members menu, then selected that cluster member from the Enabled generic server cluster members menu and click **<**.
   13) Click **OK** to return to set the time-of-day rules and continue creating your generic server cluster route configuration.

d. If you are creating a failure route, follow this additional step: In the **Fail status code** field, Eenter the status code that must be used to indicate a request was not successful.

e. If you are creating a redirection route, follow this additional step: In the **Redirect URL** field, enter the URL that must be used to redirect the inbound request.

f. If you are creating a local route, follow this additional step: Confirm that the static file document root is correct. If the value that is listed is not the document root that you want, click **Edit**. See "Administering proxy virtual hosts" on page 338 for more information on changing the static file document root.

8. Click **OK** to finish creating your proxy action rule.

Proxy server actions

Proxy server actions are used in association with proxy rule expressions. If a proxy rule expression evaluates to true, then all the proxy actions associated with the rule expression are performed. Some examples of proxy server actions include caching actions, rewriting actions, compression actions, header modification actions, and routing actions.

Caching actions

Caching actions are set to determine whether a response is cached. A caching action specifies the last modified factor and the default expiration to define how a response is cached.

Rewriting actions

Rewriting actions define how the proxy server rewrites uniform resource locators (URL). A rewriting action is used to rewrite elements of a response message. This is often done to mask the back-end server identity with that of the proxy server. The follow elements can be rewritten using this type of proxy action:

- Absolute URL response
- Redirect location header
- Redirect status code
- Relative URL response
- Set-Cookie

Compression actions

HTTP Compression actions are set to compress the request message body to the server or response message body to the client. The supported compression type standards for these proxy actions are Deflate and Gzip.

Header modification actions

Header modification actions are implemented to perform a header modify action on a specified HTTP header. The available header modify actions include:

- Set
- Append
- Edit
- Remove

The header modify action is performed only if the expression matches when the specified value is applied. For HTTP requests, the header modify action is performed on the HTTP methods you specify. For HTTP responses, the header modify action is performed on the HTTP status codes you specify.

Routing actions

Routing actions are used to route requests when a given rule expression is matched. The following types of routing actions are available:

- Application server routes

  Application server routing actions allow you to specify time of day mappings for your application servers. These mappings include or exclude an application server for the routing of requests during a specified time of day. If multiple time of day mappings are configured, the order they are matched is the same as they appear in the application server route configuration.
- Generic server cluster routes

  Generic server cluster routing actions work similar to application server routes but apply to generic server clusters instead of application servers.
- Fail routes

  Fail routing actions are used to return a failure status code to an inbound request. The value of the failure status code is specified in the fail routing action configuration.
- Redirect routes

Redirect routing actions are used to redirect an inbound request to a different URL. The URL the request is being redirected to is specified in the redirection routing action configuration.

- Local routes

  Local routing actions are used to pass an inbound request to be served by the local web applications deployed for the cell.

.

Proxy actions collection

Use this page to administer actions for the proxy server. Proxy actions include creating, modifying, or deleting rules that affect caching, compression, headers, rewriting, and routing for the proxy server. The Proxy actions collection panel allows you to configure proxy actions from one interface.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> Proxy actions**.

Caching actions

The Caching actions table contains the following fields. Caching actions are set to determine whether a response is cached.

Action name
Specifies the name of the caching action. A proxy action name must be unique within a cell and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '.

Enable caching
Specifies to enable or disable caching.

Default expiration
Specifies the default expiration, in seconds, that is used to determine the validity of cached responses for the URI that is associated with the cache rule.

Last modified factor
Specifies the period of time since the last modification.

Compression actions

Compression actions are set to compress the request message to the server or response message to the client. The Compression actions table contains the following fields.

Action name
Specifies the name of the compression action. A proxy action name must be unique within a cell and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '.

Compression action type
Specifies whether the action type is a request or a response compression action.

Compression type
Specifies the compression type standard to apply to the outbound request sent to the target server or the compression type standard to apply to the response sent to the client.

Header actions

Header actions allow you to add, modify, or delete request and response headers. The Header actions table contains the following fields.

Action name
Specifies the name of the header action. A proxy action name must be unique within a cell and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '.

Header action type
Specifies whether the action type is a request or a response header action.

Header modify action
Specifies the action to be taken on a request or response header, such as set, append, edit, or remove.

Header name
Specifies the header name to be sent in the request.

Header value
Specifies a user-defined value for a request or response header.

Rewriting actions

A rewrite action can modify inbound requests handled by the proxy server. Rewriting actions define how the proxy server rewrites the URL of a response message; for example, to mask the back-end server identity with that of the proxy server. The Rewriting actions table contains the following fields.

Action name
Specifies the name of the rewriting action. A proxy action name must be unique within a cell and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '.

Rewriting action type
Specifies a rewriting action type from a predefined list. See the following table for details.

Rewriting action type	Description
Absolute URL response	Rewrites an absolute URL from matching attributes in a response.
Redirect location header	Rewrites the URL in the relocation header in the HTTP response.
Redirect status code	Specifies the redirect status code in the first line of a response message.
Relative URL response	Rewrites a relative URL in tag attributes a response.
Set cookie domain	Rewrites the domain attribute of the set cookie header.
Set cookie path	Rewrites the path attribute of the set cookie header.

From pattern
Specifies the original URL pattern in the 302 response header from the target server. The pattern can include the asterisk (*) as a wild card symbol. A URL pattern can have one or more asterisks.

To pattern
Specifies the resulting pattern after the rewrite. The pattern can include the asterisk (*) as a wild card symbol. A URL pattern can have one or more asterisks.

Routing action

Routing actions define routes to local file system resources for static file serving. The Routing actions table contains the following fields.

Action name
Specifies the name of the routing action. A proxy action name must be unique within a cell and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '.

Routing type
Specifies a routing type from a predefined list: Application server route, Generic server route, Fail route, Redirect route, and Local route.

Note: For more detailed configuration information, see the individual proxy action settings topics.

Caching action settings

You can configure caching action settings for a proxy server. Caching actions are set to determine whether a response is cached. A caching action specifies the last modified factor and the default expiration to define how a response is cached.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> Proxy actions >** *action_name*.

Name

Specifies a user-defined symbolic name for a caching action.

A caching action name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '.

Enable caching

Specifies whether or not to enable caching for the proxy server.

You must enable caching before you can provide a value for Default expiration or Last modified factor. If a value is specified for Last modified factor, then the value for Default expiration is not used.

gotcha: If caching is disabled, the proxy server does not cache the uniform resource identifier (URI) that is associated with the proxy rule expression. For example, if a request comes in and it matches the rule expression for a proxy virtual host, then the actions associated with the rule expression, including the caching action, are executed. The rule expression may or may not include an explicit URI or URIgroup. If a request matches a rule expression with an action of caching disabled, the proxy server does not cache the response for that request.

Default expiration

Specifies the amount of time, in seconds, before a cached response expires. The default expiration determines the validity of cached responses for the URI that is associated with the cache rule.

Last modified factor

Specifies the amount of time, in seconds, before a response is cached if the response does not have explicit HTTP expiration headers. The last modified time header must be specified in the response. The value is determined based on when an HTML file was last changed to prevent caching frequently modified files for long periods of time.

HTTP compression action settings

You can configure settings for an HTTP request compression action or an HTTP response compression action for a proxy server. Compression actions are set to compress the request message to the server or response message to the client.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> Proxy actions >** *action_name*.

Name

Specifies a user-defined symbolic name for an HTTP compression action.

An HTTP compression action name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '.

Compression type

For requests, a compression type specifies the compression type standard to apply to the outbound request body sent to the target server. For responses, compression type specifies the compression type standard to apply to the response body sent to the client.

The following standards are supported: Gzip and Deflate. If you specify Any, either standard can be used.

Gzip

Deflate

Any

Content types

Specifies the content types for which compression is applied.

HTTP header action settings

You can configure settings for an HTTP request header action or an HTTP response header action for a proxy server. Use header modification actions to add, modify, or delete request and response headers.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> Proxy actions**.

Action name

Specifies a user-defined symbolic name for an HTTP header action.

An HTTP header action name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '.

Header name

Specifies the name of the HTTP header to set, append, edit, or remove from a request header or a response header.

Header modify action

Specifies the action to be taken on a request or response header, such as set, append, edit, or remove.

Table 24. Header actions.

This table lists the types of header modification actions.

Types of actions	Description
Set	Adds a new value for a request or response header. If a value does not exist, this value is added. If a value does exist, this value replaces the current value.
Append	Adds a value to the current value for a request or response header. If a value does not exist for the header, then this value is added. If a value does exist for the header, then this value is added to the end of the current value.
Edit	Changes the current value to a different value for a request or response header. If a value exists for the header, then this value is changed equivalent to a regular expression search and replace.
Remove	Removes the specified header. If the header exists, then the header is removed. If the header does not exist, then no action is taken.

Header value

Specifies a user-defined value for a request or response header.

Header value expression

Specifies a regular expression applied to the value of the Header name field. If a match exists, then the value specified for the header value field replaces the current value of the header name field.

Method names (HTTP header request action)

Specifies the HTTP method to which the action applies, such as GET or POST. If a value is not specified, then the action is applied for all method names.

Method names apply to HTTP header request actions only.

Status codes (HTTP header response action)

Specifies the return status codes to which the action applies, such as 200 or 503. If a value is not specified, then the action is applied for all status codes.

Status codes apply to HTTP header response actions only.

Rewrite action settings

You can configure settings to implement a rewrite action for outbound responses handled by the proxy server. Rewriting actions define how the proxy server rewrites elements of the uniform resource locators, such as URLs in an HTML page or the redirect link in the response. Rewrite actions are often done to mask the back-end server identity with that of the proxy server.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> Proxy actions >** *action_name*.

Action name

Specifies a user-defined symbolic name for a rewriting action.

A rewriting action name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '.

Rewriting action type

Specifies the type of rewriting action to perform. You can specify the following rewriting action types: Absolute URL response, Redirect location header, Redirect status code, Relative URL response, Set-Cookie.

Table 25. Rewriting action types.

This table lists the rewriting action types.

Action type	Description
Absolute URL response	Rewrites the absolute URI in the tag attribute in the HTTP response. The proxy server scans the response for an attribute matching the From pattern. If a match of the From pattern occurs, then the proxy rewrites the response based on the To pattern. For example: `frPattern = '/(.*)'` `toPattern = '/prefix/$1'` The tag <img src="http://someserver/1.jpg" /> is changed to <img src="http://someserver/prefix/1.jpg" />.

Table 25. Rewriting action types (continued).

This table lists the rewriting action types.

Action type	**Description**

Redirect location header	Rewrites the URI in the relocation header in the HTTP response. For example:
	`fromPattern = 'http:(.*)'`
	`toPattern = 'https:$1'`
	
	The location header: "Location: http://www.ibm.com" is changed to "Location: https://www.ibm.com."

Redirect status code	Specifies the redirect status code in the first line of a response message, such as 301 or 302.

Relative URL response	Rewrites an Relative URL in tag attributes a response. The proxy server scans the response for an attribute matching the From pattern. If a match of the From pattern occurs, then the proxy will rewrite the response based on the To pattern. For example:
	`fromPattern = '/(.*)'`
	`toPattern = '/prefix/$1'`
	
	The tag <img src="/myimages/1.jpg" /> is changed to <img src="/prefix/myimages/1.jpg" />.

Relative URL response:Passive	Instead of rewriting the response directly, the proxy server will inject a cookie in the response header. For example: If a request for *"/myimages/1.jpg"* is resent from the browser with the cookie, then the proxy server will recreate the request URI as *"/prefix/myimages/1.jpg"*. This feature requires a browser that supports cookies and for each session, only one passive rule can be defined.

Set-Cookie_Domain	Rewrites the domain attribute of the set cookie header. For example:
	`fromPattern = '(.*)'`
	`toPattern = '$1.cn'`
	
	The set cookie header: "Set-Cookie: JSESSIONID: abcdefg; domain="www.ibm.com"" is changed to be "Set-Cookie: JSESSIONID: abcdefg; domain="www.ibm.com.cn""

Set-Cookie_Path	Rewrites the path attribute of the set cookie header. For example:
	`frPattern = '(.*)'`
	`toPattern = '/prefix$1'`
	
	The set cookie header: "Set-Cookie: JSESSIONID: abcdefg; domain="www.ibm.com"; path="/"" is changed to "Set-Cookie: JSESSIONID: abcdefg; domain="www.ibm.com"; path="/prefix/"".

From pattern

Specifies the original URL pattern in the response from the target server. The pattern can include the following wild card symbol: * . A URL pattern can have one or more asterisks (*).

To pattern

Specifies the resulting pattern after the rewrite. The pattern can include the following wild card symbol: * . A URL pattern can have one or more asterisks (*).

Enable passive rewrite

Specifies whether or not to defer the rewriting of the URI until the subsequent request for that URI is sent by the client. Enabling passive rewrite prevents the proxy server from rewriting all the links in the response before sending the response back to the client.

Cookie name

Specifies the cookie for which path or domain attributes are rewritten. This setting is only valid when the action type is Set-Cookie path or Set-Cookie domain.

Limit URL pattern

Specifies to match a request URL to rewrite in the response message. Limiting the URL pattern prevents the proxy server from rewriting all URL patterns in the response message of a certain page, allowing the proxy server to skip parsing responses for other pages if there are multiple pages. This setting is only valid when the action type is absolute URL response or relative URL response.

Limit cookie domain

Specifies a constraint to limit the rewriting of the cookie domain to only a set of specified domains. If no domains are specified, then all domains are rewritten. This field is only valid when the rewriting action type specified is Set cookie domain.

Limit cookie path

Specifies a constraint which limits rewriting the cookie path to the specified paths. If no paths are specified, then all paths are rewritten. This field is only valid when the rewriting action type specified is Set cookie path.

Route action settings

You can configure settings for a route action for a proxy server. Add a route action to define routes to local file system resources for static file serving.

This topic is for local route actions, fail route actions, redirect route actions, and application server route actions. Some settings only apply to a specific type of route action. See the topic about generic server route actions for information about generic server route actions.

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere proxy servers** > *proxy_server_name* > **Proxy actions** > *action_name*.

Action name

Specifies a user-defined symbolic name for a route action.

A route action name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '.

Static file document root: /home/action1 (local route action)

Specifies the root directory on the file system where static files are located. Click **Edit** to access the Proxy Settings page to set the static file document root for a proxy server or an individual proxy server virtual host.

Static file document root applies to local route actions only.

Fail status code (fail route action)

Specifies the HTTP status code to send to the client indicating that the request failed. Possible values are integers with a value greater than zero.

Fail status code applies to fail route actions only.

Redirect URL (redirect route action)

Specifies the URL to send to the client in a 302 redirect response.

Redirect URL applies to redirect route actions only.

Time of day rules (application server and generic server cluster routes

Specifies a time rule, which is typically used to configure a maintenance window. You can specify a start time and an end time to include or exclude routing to a particular cluster member.

Click **New Time Mapping** to access the Time Mapping settings page to specify a time interval for the time mapping. You can create, delete, or edit time mappings.

Table 26. Time mapping settings.

This table lists the time mapping settings.

Setting	Description
Time	Specifies the start time and end time for the time mapping.
Action	If the action is set to include, then the server routes actions to the cluster members specified in the time mapping. If the action is set to exclude, the server does not route actions to the specified cluster members during this time interval.

Generic server cluster route action settings

You can configure a generic server cluster route action for a proxy server. Add a generic server cluster route action to define routes for inbound requests to specific generic server clusters.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> Proxy actions >** *action_name*.

Action name

Specifies a user-defined symbolic name for a generic server cluster route action.

A generic server cluster route action name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '.

Generic server cluster name

Specifies the generic server cluster name provided when the generic server cluster was created. This is a user-defined field. The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '.

The name specified must be unique among generic server clusters and cannot begin with a period or a space. A space does not generate an error, but leading and trailing spaces are automatically deleted.

Active affinity

Specifies whether the proxy server manages the application affinity to the target generic server. Active affinity is used when a generic server application cannot manage active affinity.

A cookie is used to maintain affinity and ensure that a request is returned to the correct server within the configured time. The Default expiration field specifies the amount of time, in seconds, that the active affinity set by the proxy server is valid. The cookie is automatically set to expire 24 hours after the affinity is no longer valid.

Passive affinity

Specifies whether the proxy server maintains application affinity based on a specified cookie that is set by the target generic server. Passive affinity is used when a generic server sets an affinity cookie that the proxy can use to manage affinity to a generic server.

The administrator specifies a cookie name and then maps the cookie values to each generic server.

Cookie name

Specifies the cookie that is sent by the server and used by the proxy to determine affinity.

Cookie mapping

Specifies the cookie settings to use for a particular generic server.

Table 27. Cookie mapping settings. This table lists the cookie mapping settings to use for a particular generic server.

Action type	Description
Cookie value	Specifies the value for the cookie that is set by the server. The cookie value is mapped to the server defined by the host and port.
Host	Specifies the host name for the target server.
Port	Specifies the port for the target server.

Time of day rules

Specifies a time rule, which is typically used to configure a maintenance window, for generic server cluster routes. You can specify a start time and an end time to include or exclude routing to a particular cluster member.

Click **New Time Mapping** to access the settings page to specify a time interval for the time mapping. You can create, delete, or edit time mappings.

Table 28. Time mapping settings. This table lists the time of day rules mapping settings to use for a particular generic server.

Setting	Description
Time	Specifies the start time and end time for the time mapping.
Action	If the action is set to include, then the server routes actions to the cluster members specified in the time mapping. If the action is set to exclude, the server does not route actions to the specified cluster members during this time interval.

Time mapping settings

You can configure time mapping settings for the proxy server that set routing rules to be in effect during specific time intervals. These settings can be specified for application server members or for generic server cluster members.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> Proxy actions >** *routing_action_name* > *time_mapping*.

Note: Select **Proxy Virtual Host Configuration** under Proxy Actions and create a new routing action name if you have not already configured one.

With the time mapping panel, you can make specific servers active or inactive during specified time intervals.

Start time

Specifies the start time, in hours and minutes, for when the routing action being created is valid. The time entered for this field uses a 24-hour clock.

End time

Specifies the end time, in hours and minutes, for when the routing action being created is no longer valid. The time entered for this field uses a 24-hour clock.

Action

Specifies to include or exclude the application server or generic server cluster members when a routing request is sent during the specified Start time and End time.

`Include`
: Specifies the servers to include in routing decisions during the specified time period.

`Exclude`
: Specifies the servers to exclude from routing decisions during the specified time period.

Application server members

Specifies the available application servers that can use the specified time mappings. You can add or remove application servers. The servers listed in the Enabled list are included or excluded from routing requests as selected.

Note: This field only displays if you configured an application server member.

Generic server cluster members

Specifies the available generic server clusters that can use the specified time mappings. You can add or remove generic cluster members. The servers listed in the Enabled list are included or excluded from routing requests as selected.

Note: This field only displays if you configured a generic server cluster member.

Administering custom advisors for the proxy server

You can administer custom advisors for the proxy server. Custom advisors allow for more specific determination of target application server availability by sending protocol level requests to back-end servers. Custom advisors are unique based on the combination of the business level application ID and the composition unit ID.

About this task

Complete these steps to administer actions for the proxy server.

Procedure

1. From the administrative console, select **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> Custom advisor policies**.
2. Click the name of a custom advisor to change settings.
3. Specify the business level application ID.
4. Specify the composition unit ID.
5. Specify the poll interval using seconds in the Poll interval field..
6. Specify the connection timeout using seconds in the Connect timeout field.
7. Specify the input/output timeout using seconds in the I/O timeout field.
8. Optional: Select **Enable logging** if you would like to use logging for this custom advisor. If you enable logging, then follow these steps:
   a. Select **Enable log file wrapping** if you would like wrapping turned on.

   b. Specify the maximum size of the log file using megabytes in the Log file size field.
9. Create mappings to be used by your custom advisor using one of the following sub-steps:
   a. "Configuring stand-alone application server mappings" on page 332
   b. "Configuring application server cluster mappings" on page 334
   c. "Configuring generic server cluster mappings" on page 335
10. Optional: Click **Delete** to remove a selected mapping from the collection.

Custom advisor policies

Custom advisor policies help to determine target application server availability. Custom advisors are Java code modules that work within the proxy server to provide information about the application's availability to the proxy server selection code.

Custom advisor policies provide a mechanism to interpret an application protocol response message to determine if an application server, a cluster, or a generic server cluster should be used by the proxy server when requests are made. A custom advisor is able to verify that the application is available, functioning properly, and has access to all required resources.

The proxy server periodically performs an advisor cycle. During an advisor cycle it will call the isUsable() method defined on the custom advisor for each available application server, cluster, and generic server cluster targeted by that custom advisor. The isUsable() method passes an AdvisableServer object, which is used to determine the address, port, and protocol for the targeted application servers, clusters, and generic server clusters.

The custom advisor sends a request to the targets and if communication is successful, then receives their responses. Using the responses, the custom advisor determines if the target is usable or is not usable. If the target is not usable, then the target will be marked as unavailable and will not be used for selection. A target that is marked as unavailable will not be used for selection for this application again, until it is determined to be available by a future advisor cycle. If the target is determined to be usable, a value of true is returned, and the target will continue to be used for selection.

Custom advisors collection

Use this page to administer custom adviser policies. With custom advisors, you can determine the availability of a specific target application server by sending protocol level requests to back-end servers. Custom advisor policies are unique based on the combination of the business-level application ID and the composition unit ID.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> Custom advisor policies**.

Business-level application ID

Specifies a unique identifier for a business level application. A business level application is a configuration that is stored in the product configuration repository.

Composition unit ID

Specifies a unique identifier for a composition unit. A composition unit is a registered custom adviser asset that has additional configuration information, which you specified when adding the asset to the application.

Custom advisor policy settings

You can configure settings for a custom advisor policy. Custom advisor policies allow for more specific determination of target application server availability by sending protocol level requests to back-end servers. Custom advisor policies are unique based on the combination of the business-level application ID and the composition unit ID.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> Custom advisor policies >** *bla_id*.

Business-level application ID

Specifies a unique identifier for a business level application. A business-level application is a configuration that is stored in the product configuration repository.

Composition unit ID

Specifies a unique identifier for a composition unit. A composition unit is a registered custom advisor asset that has additional configuration information, which you specified when adding the asset to the application.

Poll interval

Specifies the amount of time, in seconds, between requests for custom advisors.

Connect timeout

Specifies the amount of time, in seconds, to wait before a connection attempt fails.

I/O timeout

Specifies the amount of time, in seconds, for a custom advisor to wait before read and write operations succeed. If the timeout is reached, then the server stops.

Enable logging

Specifies whether or not logging is enabled. By default, the check box is cleared.

If logging is enabled, then the proxy server logs error messages to a log file. If logging is not enabled, log messages are not added to a log file.

Enable log file wrapping

Specifies whether information at the beginning of the log file is overwritten when the maximum log file size is reached. The default value is True.

Log file size

Specifies the size, in megabytes, of the log file.

Custom advisor mappings

Specifies the application servers, application server clusters, or generic server clusters the custom advisor monitors.

Click **New custom advisor mapping** to access the Custom advisor settings page to create a new custom advisor mapping for a custom advisor.

Configuring stand-alone application server mappings

You can configure a stand-alone application server mapping. Stand-alone application server mappings specify that a stand-alone application server is mapped to a custom advisor.

Before you begin

Stand-alone application server mappings cannot be configured until a proxy server has been configured and a custom advisor has been deployed.

About this task

Complete these steps to configure a stand-alone application server mapping.

Procedure

1. From the administrative console, select **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> Custom advisors policies >** *custom_advisor_namestandalone_application_server_mapping_name*.
2. Click **New Stand-alone Application Server Mapping** to begin configuring a new stand-alone application server mapping.
3. In the Cell name menu, select the cell where the server is located that the custom advisor will monitor.
4. In the Node name menu, select the node where the server is located that the custom advisor will monitor. Only the nodes contained within the cell you selected will be available in the Node name menu.
5. In the Server name menu, select the server the custom advisor will monitor. Only the server located on the node you selected will be available in the Server name menu.
6. Specify the application to be monitored. Only the applications deployed on the server you selected will be available in the Application name menu.
7. From the menu, choose a transport chain.
8. Click **OK**.

Stand-alone application server cluster mapping settings

You can configure settings for a stand-alone application server cluster mapping. Stand-alone application server cluster mappings are specified only for a stand-alone application server cluster custom advisor.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> Custom advisors >** *custom_advisor_name > standalone_application_server_cluster_mapping*.

Cell name:

Specifies a logical name for the cell.

A cell name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '.

Node name:

Specifies the name of the node. A node is a logical grouping of managed servers.

A node name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '.

Server name:

Specifies the display name for the server.

A server name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '.

Application name:

Specifies the name of the application.

An application name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '.

Transport chain:

A transport chain consists of one or more types of channels, each of which supports a different type of I/O protocol, such as TCP, HTTP, or DCS.

Transport chains enable communication through transport channels, or protocol stacks, which are usually socket based.

Configuring application server cluster mappings

You can configure an application server cluster mapping to specify which application server clusters a custom advisor will monitor.

Before you begin

Application server cluster mappings cannot be configured until a proxy server has been configured and a custom advisor has been deployed.

About this task

Complete these steps to configure an application server cluster mapping.

Procedure

1. From the administrative console, select **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> Custom advisors policies >** *custom_advisor_nameapplication_server_cluster_mapping_name*.
2. Optional: Click **New Application Server Cluster Mapping** to begin configuring a new application server cluster mapping.
3. In the Cell name menu, select the cell of the server that the custom advisor will monitor.
4. In the Cluster name menu, select the cluster of the server that the custom advisor will monitor. Only the available clusters within the cell you selected are available in the Cluster name menu.
5. Specify the application to be monitored. Only the applications that are deployed on the cluster you selected are available in the Application name menu.
6. From the menu, choose a transport chain.
7. Select the cluster members that the custom advisor will monitor from the Available application server cluster members menu.
8. Click **>** to add the selected cluster members to be monitored by the custom advisor.
9. Optional: If you want to remove a cluster member from the Selected application server cluster members menu, select the cluster members to remove and click **<**.
10. Click **OK**.

Application server cluster mapping settings

You can configure settings for an application server cluster mapping. Application server cluster mappings are specified only for an application server cluster custom advisor.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> Custom advisors***custom_advisor_name* > *application_server_cluster_mapping*.

Cell name:

Specifies a logical name for the cell.

A cell name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '.

Cluster name:

Specifies a logical name for the proxy cluster.

A cluster name must be unique among proxy clusters within the containing cell and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '.

Application name:

Specifies the name of the application.

An application name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '.

Transport chain:

A transport chain consists of one or more types of channels, each of which supports a different type of I/O protocol, such as TCP, HTTP, or DCS.

Transport chains enable communication through transport channels, or protocol stacks, which are usually socket based.

Application server cluster members:

Specifies the application server clusters that use the specified cluster mapping. You can add or remove application server cluster members.

An application server cluster is a cluster of application servers. Application servers host a common set of resources and can receive routing actions as a unit.

Node name
Specifies the name of the node. A node is a logical grouping of managed servers. A node name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '.

Server name
Specifies the display name for the server. A server name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '.

Configuring generic server cluster mappings

You can configure a generic server cluster mapping. A generic server cluster allows you to configure external servers into a logical cluster that can be used by the proxy server to route requests. Generic servers host a common set of resources and can receive routing actions as a unit. Generic server cluster mappings are configured to specify which generic server clusters a custom advisor will monitor.

Before you begin

Generic server cluster mappings cannot be configured until a proxy server has been configured and a custom advisor has been deployed.

About this task

Complete these steps to configure a generic server cluster mapping.

Procedure

1. From the administrative console, select **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> Custom advisors policies >** *custom_advisor_namegeneric_server_cluster_mapping_name*.
2. Optional: Click **New Generic Server Cluster Mapping** to begin configuring a new generic server cluster mapping.
3. In the Cluster name menu, select the generic server cluster the custom advisor will monitor is located.
4. Select the generic cluster members the custom advisor will monitor from the Available generic server cluster members menu.
5. Click **>** to add the generic server cluster members to be monitored by the custom advisor.
6. Optional: If you want to remove a generic cluster member from the Selected generic server cluster members menu, then select the generic cluster member to remove and click **<**.
7. Click **OK**.

Generic server cluster mapping settings

You can configure settings for a generic server cluster mapping. Generic server cluster mappings are specified only for a generic server cluster custom advisor.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> Custom advisors >** *custom_advisor_name* > *generic_server_cluster_mapping*.

Cluster name:

Specifies a logical name for the proxy cluster.

A cluster name must be unique among proxy clusters within the containing cell and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '.

Generic server cluster members:

Specifies the generic server clusters that use the specified time mappings. You can add or remove generic server cluster members.

A generic server cluster is a cluster of generic servers. Generic servers host a common set of resources and can receive routing actions as a unit. Generic servers, such as web servers, are not managed by the application server.

Host name
Specifies the IP address, domain name system (DNS) host name with domain name suffix, or the DNS host name for the generic server cluster.

Port
Specifies the port where the generic server cluster accepts client requests. Specify a port value in conjunction with the host name.

Creating custom advisors for the proxy server

Use the following steps to create and deploy a custom advisor to a proxy server. Custom advisor policies allow for more specific determination of target application server availability. Custom advisors are Java code modules written that work within the proxy server to provide information about application server availability to the proxy server selection code.

About this task

Complete these steps to create and deploy a custom advisor for the proxy server.

Procedure

1. Create the custom advisor code.

   Custom advisors are written in the Java language. A custom advisor extends the com.ibm.wsspi.advisor.AbstractCustomAdvisor class in the proxy.jar file that is included with WebSphere Application Server. Custom advisors use the defined methods of the AbstractCustomAdvisor class to obtain the information about the advisor. A custom advisor also must implement the following elements:

   - A constructor method that takes a CustomAdvisorConfigObject object and calls the super(caConfigObject) method, for example

```
public AbstractHttpProxyCustomAdvisor(CustomAdvisorConfigObject caConfigObject) {
  super(caConfigObject);
 }
```

   - An isUsable() method that takes an AdvisableServer object; for example

```
public boolean isUsable(AdvisableServer aServer) throws CustomAdvisorException;
```

   The initialize method is called after the AbstractCustomAdvisor construction, but before the isUsable method is called. This process allows the custom advisor to perform any additional steps after the base class completes initialization, but before the isUsable method is called, which ensures that the initialize method is only called once. If overridden, then the initialize() method must call the super.intitialize method, for example:

```
protected void initialize() {
     super.initialize();
    }
```

   For more information about the required routines and the other methods available to a custom advisor, see the application programming interface (API) reference section of the information center. From the information center navigation, scroll to the Reference section and click APIs - Application Programming Interfaces. A list of the product API specifications displays in alphabetic order.

   There are two exception classes that need to be considered when creating the custom advisor:

   - The CustomAdvisorException can be created by the isUsable method of the custom advisor to tell the AbstractCustomAdvisor that the custom advisor must not call the isUsable method again until the next advisor cycle.
   - The NoLogConfiguredException is created by the AbstractCustomAdvisor if the no logging file has been configured for the custom advisor, but logging is enabled.

   The `httpcustomadvisor.jar` file can be used as a sample of a custom advisor. This file contains an AbstractHttpProxycustomAdvisor.java class that extends the com.ibm.wsspi.advisor.AbstractCustomAdvisor and implements the isUsable() and initialize() methods.

   Sample `httpcustomadvisor.jar` file:

```
<advisor-context>
<description>Webbsphere Proxy Demo HTTP Advisor Context</description>
<display-name>Webbsphere Proxy Demo HTTP Advisor Context</display-name>
?
<advisor>
<advisor-name>WebsphereProxyDemoHttpAdvisor</advisor-name>
?
<advisor-class>
com.ibm.ws.proxy.demo.customadvisor.http.HttpProxyCustomAdvisor
</advisor-class>
<description>Demo Websphere Proxy Http Advisor
Implementation</description>
<display-name>Demo Websphere Proxy Http Advisor</display-name>
</advisor>
</advisor-context>
```

2. Compile your custom advisor code. After you have created the Java source code for your custom advisor, you must compile it using the AbstractCustomAdvisor code that is included with WebSphere

Application Server. To access the abstract custom advisor classes in the com.ibm.wsspi.advisor package, add the proxy.jar file to your Java class path. The proxy.jar file is located in the ${WAS_INSTALL_ROOTl/plugins directory.

3. Create the advisor-context.xml file.

   After compiling the custom advisor code, you will need create the advisor-context.xml file. This file is used to identify the code as a custom advisor Java archive (JAR) file when it is imported as an asset and then added as a compilation unit to a business-level application (BLA). When the custom advisor JAR asset is added to a BLA and then targeted to a proxy server, the Content DistributionFramework (CDF) support will distribute and copy all the BLA artifacts of the custom advisor to the appropriate configuration information on the targets specified.

   The advisor-context.xml contains the class name for the custom advisor to be run and the custom advisor name. The format of the advisor-context.xml file must follow the advisor-contex.xsd schema in the proxy.jar file. You can use an XML schema tool to assist in creating and setting the appropriate information. The required configuration information is defined as follows:

   ```
   <advisor-name> SomeCustomAdvisor </advisor-name>
   <advisor-class> com.ibm.wlm.test.customadvisor.SomeCustomAdvisor </advisor-class>
   <description> Some Custom Advisor Description </description>
   <display-name> Some display name </display-name>
   ```

4. Create the custom advisor BLA. Package your compiled custom advisor class files and the advisor-context.xml into a JAR file. This JAR file is then used when creating the custom advisor BLA to be installed and deployed to the proxy server. The following example shows the commands to use to install a custom advisor as a BLA.

```
$AdminTask importAsset {-source C:/proxy/testadvisor.jar -storageType FULL}
$AdminTask createEmptyBLA {-name myBLA}
$AdminTask addCompUnit {-blaID myBLA -cuSourceID assetname=testadvisor.jar,assetversion=1.0 -MapTargets {{.* ProxyServer}} -CustomAdvisorCUOptions
{{"type=Cluster,cellName=yourCellName,clusterName=yourClusterName
,applicationName=myBLA" default default default 1000}}}
$AdminConfig save
```

5. Configure your deployed custom advisor. See “Administering custom advisors for the proxy server” on page 330 for additional details.

Administering proxy virtual hosts

Virtual hosting allows a single proxy server to host multiple domains and ports on a single IP address and port. A proxy virtual host can be created for each Web domain the proxy server is hosting, or wild card characters can be used to host multiple Web domains with a single proxy virtual host.

About this task

Note: A proxy virtual host allows a single proxy server to host multiple domains and ports on a single IP address and port. A proxy virtual host consists of the name and the port representing the Web domain and a set of proxy rule expressions to perform specified proxy actions when a defined criteria exists. Additionally, each proxy virtual host can override the server scope configuration of the proxy server to have configuration elements defined specifically for that virtual host. Proxy virtual hosts use a set of proxy server actions and proxy rule expressions. Proxy rules expressions are evaluated when inbound requests are received by the proxy virtual host. If the expression is evaluated to be true, any proxy server actions specified by the proxy rule expression are performed

Complete these steps to administer or create a new proxy virtual host.

Procedure

1. From the administrative console, select **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> Virtual hosts >** *virtual_host_name*.
2. Optional: Click **New** to access the virtual hosts settings page to configure settings for a new virtual host.

3. Specify a name for the virtual host. The name of the virtual host must match the Web domain it represents. If the web domain is www.proxy1.com, the name of the proxy server host must be www.proxy1.com. The asterisk symbol, * , can be used as a wild card character to represent all Web domains. If the proxy virtual host is *:80, then all inbound requests on port 80 are handled by that proxy virtual host regardless of what Web domain is being requested.
4. Specify the port for the virtual host. The port of the virtual host must match the port used by the web domain it represents. If the Web domain uses port 80, then the port of the proxy virtual host must also be 80. The asterisk symbol, * , can be used as a wild card character to represent all ports. If the proxy virtual host is www.proxy1.com:*, then all inbound requests to www.proxy1.com are handled by that proxy virtual host regardless of what port is being used.
5. Choose one or more proxy rule expressions for the virtual host. Proxy rule expressions allow for proxy actions to be performed when the expression evaluates to true. See "Proxy rule expressions" on page 343 and "Proxy server actions" on page 319 for more information on proxy rule expressions and proxy server actions.
6. Optional: Move a proxy virtual host up or down in the list to ensure the correct proxy virtual host is used. The usage of wild card characters for virtual proxy hosts creates a situation where an inbound request might match multiple proxy virtual hosts. In this scenario, the request will be handled by the first proxy virtual host that matches the request.
7. Optional: Click **Edit** to change the selected proxy rule expression.
8. Optional: Click **Proxy Virtual Host Settings** to override the server scoped settings for static file serving, logging, or error page policy.
   a. If you want to override the static file serving settings for this proxy virtual host, click **Static File Serving**, select **Customize for this virtual host**, and then specify a new value for the **Static file document root** field.
   b. If you want to override the logging settings for this proxy virtual host, click **Logging**, select **Customize for this virtual host**, and then specify new values for one or more of the following fields:
      - **Enable access logging**
      - **Access log maximum size**
      - **Proxy access log**
      - **Cache access log**
      - **Local access log**
   c. If you want to override the error page policy settings for this proxy virtual host, click **Error Page Policy**, and then select **Customize for this virtual host**, and then specify new values for one or more of the following fields:
      - **Error page generation application URI**
      - **Handle errors generated by the proxy server**
      - **Handle errors generated by the application server**
      - **Headers to forward to error page application**
      - **HTTP status codes that are to be recognized as errors**
   d. Click **OK** to save your proxy virtual host settings, and return to the previous administrative console page.
9. Click **OK** to save all of your other changes.

Proxy virtual hosts

Virtual hosting allows a single proxy server to host multiple domains and ports on a single IP address and port.

A proxy virtual host consists of the name and port of a web domain and a set of proxy rule expressions that perform proxy actions for specific criteria. Additionally, each proxy virtual host can override the proxy server configuration to have configuration elements defined specifically for that virtual host. The following settings can specify virtual host settings in place of the server scope settings:

- Logging
- Custom error pages
- Static file serving

Proxy virtual hosts use proxy server actions and proxy rule expressions. Proxy rule expressions and proxy server actions are only used for proxy virtual hosts. When the proxy virtual host receives inbound requests, the proxy rule expressions are evaluated. If the expression is evaluated to be true, any proxy server actions that are specified by the proxy rule expression are performed. The following proxy server actions can be specified when an expression evaluates to true:

- Routing rules
- Caching rules
- URL rewriting rules
- Header modification rules
- Compression rules

A different proxy virtual host can be created for the proxy server to represent each web domain that the proxy server is hosting. For example, a request for www.proxy1.com on port 80 uses the configuration specified for www.proxy1.com:80. A request for www.proxy2.com on port 80 uses the configuration specified for www.proxy2.com:80. You can use a wild card character to specify that a proxy virtual host can be used for all web domains or all ports. For example, www.proxy1.com:* specifies that a proxy virtual host can be used for all requests for the web domain www.proxy1.com regardless of the port. A proxy virtual host for *:80 specifies that it can be used for all requests on port 80 regardless of the web domain.

After creating a proxy server with the needed proxy virtual hosts, the HTTP protocol allows multiple web domains to be hosted by a single server process. When an inbound request is received by the proxy server, it matches the proxy virtual host located in the inbound request message to the appropriate configuration for that proxy virtual host. If a request matches multiple proxy virtual hosts because wild card characters were used, the proxy virtual host that is first in the list of proxy virtual hosts is used.

Proxy virtual hosts collection

Use this page to administer proxy virtual hosts. A proxy virtual host allows a single proxy server to host multiple domains or ports on a single IP or port. Each proxy virtual host consists of a domain name and a port.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> Proxy virtual hosts**.

Virtual host name

Specifies a user-defined symbolic name for a proxy virtual host.

A virtual host name must be unique and cannot contain an invalid character. The virtual host name is a reflection of the web domain it represents. For example, use a virtual host name of www.proxy1.com to represent the www.proxy1.com web domain. The asterisk symbol, *, can be used in the name field to indicate that this proxy virtual host matches all web domains. The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '.

Proxy rule expressions

Specifies a rule expression for a proxy virtual host. A proxy rule expression consists of operands and operators that is defined by an administrator. If a proxy rule expression evaluates to true, then the proxy actions that are configured for the rule are run.

Enabled

Specifies whether or not a proxy virtual host is enabled.

If a proxy virtual host is enabled, then the configuration for the proxy virtual host is used when a user makes a request for the proxy virtual host. Otherwise, another matching proxy virtual host configuration is used. If no other proxy virtual host configuration matches the configuration for the requested proxy virtual host, then the server-scoped configuration is used.

Proxy virtual host settings

You can configure settings for a proxy virtual host. A proxy virtual host allows a single proxy server to host multiple domains or ports on a single IP or port. Each proxy virtual host consists of a domain name and a port.

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere proxy servers** > *proxy_server_name* > **Virtual hosts** > *virtual_host_name*.

Virtual host name

Specifies a user-defined symbolic name for a proxy virtual host.

A virtual host name must be unique and cannot contain an invalid character. The virtual host name is a reflection of the web domain it represents. For example, use a virtual host name of www.proxy1.com to represent the www.proxy1.com web domain. The asterisk symbol, *, can be used in the name field to indicate that this proxy virtual host matches all web domains. The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '.

Virtual host port

Specifies the port number associated with the web domain that the proxy virtual host represents. The asterisk symbol, *, can be used in the virtual host port field to indicate that this proxy virtual host matches requests for all ports of the web domain that are specified by the Virtual host name field..

Proxy rule expressions

Specifies one or more rule expressions for a proxy virtual host. At run time, the proxy host evaluates the expressions according to the order that they are defined. If the value of a proxy rule expression is `true`, the actions associated with the expression is performed by the proxy virtual hosts. Click **Edit** to change a proxy rule expression. Proxy rule expressions are only available for proxy virtual hosts.

You can enable proxy rule expressions to complete certain proxy actions for a proxy virtual host. Click the arrows to include or exclude proxy rule expressions. Proxy rule expressions are evaluated in the order that they are listed. You can move a proxy rule expression up or down in the list to determine when the rule expression is evaluated.

Available proxy rule expressions
Specifies the rule expressions that can be added to a virtual host.

Proxy rule expressions
Specifies the rule expressions that were added to the virtual host.

Proxy virtual host settings details

Use this panel to override some proxy server scoped settings with values to be used for the proxy virtual host.

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere proxy servers** > *proxy_server_name* > **Virtual hosts** > *virtual_host_name*> **Proxy virtual host settings**.

General Properties

The three types of configurations available on this page are static file serving, logging, and error page policy. For each of these types, you can choose to use the server scoped value for this setting or you can override the settings with new values that are used only for this proxy virtual host.

- If you choose **Use global server settings**, the server scoped values for that type are displayed and those values will be used for the proxy virtual host.
- If you choose **Customize for the virtual host**, the values that you enter in the fields for that type are used for the proxy virtual host instead of the server scoped values.

Static file serving

This type of configuration specifies the information needed for the proxy server to perform static file serving.

- **Static file document route**: The location on the file system of the static files to be served

Logging

This type of configuration specifies the logging options for proxied requests and stored cache requests.

- **Enable access logging**: Specifies whether or not access logging is used.
- **Access log maximum size**: Specifies the maximum size of each access in megabytes.
- **Proxy access log**: Specifies the log that is used used to log responses that are received from remote servers.
- **Cache access log**: Logs responses that are served from the local cache.
- **Local access log**: Logs all non-cache local responses, for example, redirects and internal errors.

Error page policy

This type of configuration specifies settings to support the use of customized error pages to be displayed when errors occur during the processing of requests.

- **Error page generation application URI**: If a valid URI to an installed application is not provided, the custom error page policy does not handle requests.
- **Handle errors generated by the proxy server**: Specifies if errors generated by the proxy server are handled with the custom static error pages stored on the local file system. If this is not selected, then the default error messages are used instead of any customer error pages.
- **Handle errors generated by the application server**: Specifies if errors generated by the application are handled with the custom static error pages stored on the local file system. If this is not selected, then the default error messages are used instead of any customer error pages.
- **Headers to forward to error page application**: Specifies a list of the headers from the original request to forward to the error page generation application.
- **HTTP status codes that are to be recognized as errors**: Specifies a list of the status codes in a response to direct to the error page generation application.

Administering proxy rule expressions

You can administer proxy rule expressions to make proxy actions more granular in scope. Proxy rule expressions are configurations assigned to a proxy virtual host.

About this task

A proxy rule expression consists of operands and operators that are defined by an administrator. When an expression evaluates to true, the proxy action rules associated with that proxy rule expression are performed.

Complete these steps to create a new proxy rule expression or to administer an existing proxy rule expression.

Procedure

1. From the administrative console, select **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name* **> Proxy Virtual Host Configuration > Proxy rule expressions**.
2. Click **New** to create a new proxy rule expression. To administer an existing proxy rule expression, Click ***rule_expression_name***.
3. Specify the name for the rule expression.
4. Click **Subexpression builder** to add an expression. You can also create the expression by entering the expression manually. The Subexpression builder is only designed to generate a subset of the available options when creating an expression. For example the Subexpression builder does not allow the usage of parenthesis in an expression. If a complex expression is needed, then type the expression into the field instead of using the Subexpression builder. For some examples of valid expressions, see "Proxy rule expressions." To use the Subexpression builder to create your expression, repeat these steps for each subexpression that you want to create.
   a. From the **Logical operator** menu, choose a logical operator. If this the first subexpression in this field, then the logical operator will be removed when it is generated.
   b. From the **Select operand** menu, choose an operand.
   c. From the **Operator** menu, choose an operator.
   d. From the **Select *<operand_name>*** menu, choose the name of the operand to be used. The name and available selections for this menu is dependant on the operand specified in step b. For example, if you selected **cell** in step b, then the name of this menu will **Select cell** and the menu will list the available cells as menu options.
   e. Click **Generate subexpression** to add the new expression to the Subexpressions list.
   f. Click **OK**.
5. Enable a proxy action for the proxy rule expression. You can enable multiple proxy actions. All the enabled proxy actions will be performed when the proxy rule expression evaluates to true.
6. Optional: If the a proxy action needs to be modified, select that proxy action, and click **Edit**.
7. Optional: Move a proxy action up or down in the list to change the order in which the proxy actions will be performed. If the sequence the proxy actions are performed is important for the rules you specified, ensure that the order of the proxy actions in the list corresponds to the order they must be performed.
8. Click **OK**.

Proxy rule expressions

Use proxy rule expressions to make configuration rules more granular in scope by specifying information within the rule. Proxy rule expressions are assigned to a proxy virtual host. When a request is handled by the proxy virtual host, the proxy rule expressions associated with that proxy virtual host are evaluated. If any of the proxy rule expressions evaluates to true, then all the proxy actions specified in the proxy rule expression configuration are performed.

Proxy rule expressions have operands and operators that are created and managed by an administrator. With operands, you can configure proxy rule expressions based on the following criteria:

- cell
- application
- module
- uri
- urigroup

Operands are combined with operators to define the proxy rule expressions. Operators can be applied with words or symbols.

Word	Symbol
AND	&&

Word	Symbol
OR	||

Example 1

In the following example, the expression evaluates to true if the target cell name is *mycell*. Otherwise, the expression evaluates to false.

```
cell=mycell
```

Example 2

In the following example, the expression evaluates to true if the target cell name is *mycell*, and the application name is *myapp*. Otherwise, the expression evaluates to false.

```
cell=mycell AND application="myapp"
```

Example 3

In the following example, the expression evaluates to true if the target cell name is *mycell*, and the target application name is *myapp1* or *myapp2*. Otherwise, the expression evaluates to false.

```
cell=mycell && (application="myapp1" || application="myapp2")
```

Example 4

In the following example, the expression evaluates to true if the target cell name is *mycell*, and the target application is not named *myapp* Otherwise, the expression evaluates to false.

```
cell=mycell AND application!=myapp
```

Example 5

In the following example, the expression evaluates to true if the request URI matches the pattern /proxy1/*. Otherwise, the expression evaluates to false.

```
uri="/proxy1/*"
```

Proxy rule expressions collection

Use this page to administer proxy rule expressions for the proxy server. Proxy rule expressions allow you to make configuration rules more granular in scope by specifying information within the rule. A proxy rule expression is associated with a virtual proxy host and consists of operands and operators that are defined by an administrator. When an expression evaluates to true, the proxy action rules associated with that proxy rule expression are performed.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers >***server_name* **> Proxy Virtual Host Configuration > Proxy rule expressions**.

Expression name

Specifies a user-defined symbolic name for a proxy rule expression.

An expression name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '.

Expression

Specifies the expression that is evaluated when an inbound request is made to the proxy virtual host.

An expression evaluates to true or false and consists of operands and operators. Expressions can be entered manually on the Proxy rule expressions settings page. The **Subexpression builder** can also be used on that page to generate the expressions.

Table 29. Rule expression operands.

This table lists the rule expression operands.

Operands	Description
Cell	Specifies the cell that contains the back end server to which the request is mapped.
Application	Specifies the Java Platform, Enterprise Edition (Java EE) application that a given request maps to on the back end server.
Module	Specifies the J2EE application module to which a given request is mapped.
URI	Specifies the URI for an inbound request.
URIgroup	Specifies a group of URIs to match against the URI. If the inbound request is in the URI group, then the expression evaluates to true.

Proxy actions

Specifies the proxy actions configured for the rule expression. Proxy actions are performed when the proxy rule expression evaluates to true.

You can configure the following proxy actions for a proxy rule expression:

- header actions
- caching action
- compression actions
- rewriting actions
- routing actions

Proxy rule expression settings

You can configure settings for a proxy rule expression. A proxy rule expression consists of operands and operators that are defined by an administrator. If a proxy rule expression evaluates to true, then all proxy actions configured for that rule expression are run.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers >** *proxy_server_name***Proxy Virtual Host Configuration > Proxy rule expressions >** *expression_name*.

Expression name

Specifies a user-defined symbolic name for a proxy rule expression.

An expression name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '.

Expression

Specifies the expression that is evaluated when an inbound request is made to the proxy virtual host. You can create expressions manually by entering the text into this field, or you can use the **Subexpression builder** to generate a new expression.

Specify the following settings for the **Subexpression builder**. You can use the following operands in your expression:

Table 30. Rule expression operands.

This table lists the rule expression operands.

Operands	**Description**
Cell	Specifies the cell that contains the server to which the request is mapped.
Application	Specifies the J2EE application that a given request maps to on the back end server.
Module	Specifies the J2EE application module to which a given request is mapped.
URI	Specifies the URI for an inbound request.
URIgroup	Specifies a group of URIs to match against the URI. If the inbound request is in the URI group, then the expression evaluates to true.

Proxy actions

Specifies the proxy actions configured for the rule expression.

You can configure the following proxy actions for a proxy rule expression:

- header actions
- caching action
- compression actions
- rewriting actions
- routing actions

Click **Edit** to modify a proxy action from this panel. Click the arrows to include or exclude proxy actions. You can move a proxy action up or down in the list to determine when the proxy action is performed.

Creating a custom filter and deploying it to a proxy server

You can create a custom filter for your proxy server if you need to have the proxy server perform a function, such as customized logging, that is not provided through normal product settings.

Before you begin

- Determine where you are going to store the artifact that contains the filter you create. You can import this artifact from either a local or remote file system.
- You must know the name of the proxy server on which you want to install the custom filter.
- You must start the administrative console that is used to control this proxy server, it is not already started.

About this task

A filter provides an optional, secondary control over a function, that is beyond the control that is normally provided through typical product settings. For example, an applications might use a logging filter mechanism to suppress all of the events that have a particular message key.

Procedure

1. Create a custom filter.
   a. Create a class that extends com.ibm.wsspi.HttpDefaultFilter.

b. Override the abstract doFilter method, and, optionally, the init and destroy methods. If you override the init and destroy methods, you must call the super versions of these methods to preserve filter lifecycle functionality.

c. Write a new doFilter method, that uses the HttpProxyServiceContext interface.

   This new doFilter method can be written such that it:
   - Changes any artifacts that are associated with the request or the response, such as the response itself, the response code, or the headers.
   - Changes any artifacts that are associated with both the request and the response.
   - Obtains information about either the request or the response.
   - Obtains information about both the request and the response.

d. Use the proxy server filter managed bean (MBean), proxyFilterMbean, to determine the correct ordinal for the filter.

```
proxyFilterMbean = AdminControl.queryNames('type=ProxyServerFilterBean,*')
AdminControl.invoke(proxyFilterMbean, 'viewAllFilters')
```

   The order, in which all of the filters on the proxy server are processed, displays.

e. Create the filter descriptor file, filter-context.xml

   This file is used to define all of the filters that have this descriptor. The following example illustrates a basic version of a filter-context.xml file.

   In this example. the descriptor determines where and when the given filter is executed on both the request and the response. The filter point determines where the filter is executed. In this example, the filter executes after the request is received. The ordinal determines when the filter gets processed relative to other filters at the same filter point. The higher the ordinal, the higher the filter is in the processing queue.

```
<?xml version="1.0" encoding="UTF-8>
<filter-context xmlns="http://www.ibm.com/2003/FilterContextSchema">
   <description>Proxy Filter Sample descriptor</description>
   <display-name>Proxy Sample Filter</display-name>

   <filter>
          <filter-name>HttpRequestFilter</filter-name
          <filter-class>com.ibm.ws.proxy.sample.HttpRequestFilter</filter-class>
          <description>HTTP sample filter to execute at REQUEST filter point</description>
          <display-name>HTTP Request Sample</display-name>
          <protocol-name>HTTP</protocol-name>
                <filter-point>RequestReceived</filter-point>
          <ordinal>1000</ordinal>
   </filter>
</filter-context>
```

f. Bundle the compiled .class files for the filter, and the filter descriptor into a JAR file.

2. Import the artifact (JAR file) that contains the custom filter.
   a. In the administrative console, click **Applications > Application Types > Assets > Import**.
   b. Select either **Local file system** or **Remote file system** to indicate where the JAR file is located.
   c. Specify the fully qualified name of the JAR file.

      The fully qualified name of the JAR file includes the directory path to where the file is located, and the file name. If you do not know the fully qualified name of the JAR file, you can use the browse function to locate the file.
3. Create a business level application (BLA) that includes this artifact.
   a. In the administrative console, click **Applications > New Application > New Business-level application**.
   b. In the **Name** field, specify a name for the new application that you are creating.

      You can also specify a description of the application in the **Description** field.
   c. Click **Apply**.
4. Create a BLA composition unit (CU) from the artifact.

a. In the **Deployed assets** table, click **Add > Add asset**.
b. Select the name of the artifact that you imported in the first step, and then click **Continue**.
c. Change the composition unit settings as needed, and then click **Modify target**.
d. Select the proxy server on which you want to deploy this CU from the list of available deployment targets, and then click **OK**.

   When you click **OK** the product maps the composition unit to the selected proxy server.
e. Specify the relationship options for this composition unit.
f. Click **Finish**.

To verify that the product successfully added the created this CU, click **Applications > Business-level applications >** *application_name*. If the product successfully adds the CU, the name of the CU is shown in the list of deployed assets for this BLA.

5. Start the proxy server.
6. Start the BLA that contains the filter.

Results

The filter is running on the proxy server.

What to do next

Use the proxy server filter MBean, proxyFilterMbean, to verify that the filter is installed on the proxy server and that is is being processed in the correct order relative to the other filters that are deployed on the proxy server. If you need to change the order in which this filter is processed, run the modifyOrdinal command against the proxyFilterMbean MBean.

Configuring denial of service protection for the proxy server

You can configure a pair of properties on your proxy server or DMZ Secure Proxy Server for IBM WebSphere Application Server to limit your risk against denial of service attacks involving the buffering of large HTTP payloads.

Before you begin

Denial of service protection for the proxy server or the DMZ Secure Proxy Server for IBM WebSphere Application Server is not done during the creation of these servers. The proxy server or the DMZ Secure Proxy Server for IBM WebSphere Application Server that will include denial of service protection must already exist before following these steps.

About this task

Note:

> Protection is now included to guard against a type of security breach known as a denial of service attack. This type of attack can typically send more traffic to a network address than the data buffers were designed to accommodate. The proxy server and DMZ Secure Proxy Server for IBM WebSphere Application Server have several properties that can be configured to limit your risk against denial of service attacks involving the buffering of large HTTP payloads.

A denial of service attack is a malicious type of security breach to a computer system that does not usually result in the theft of information or other security loss. This type of attack can typically send more traffic to a network address than the data buffers were designed to accommodate resulting in a loss of memory. HTTP allows for the body of a message to be sent to an HTTP server as an HTTP request or an HTTP response. The body can be sent to the HTTP server in a series of sequential network writes instead of being sent in one large network write. This is process is known as Transfer-Encoding chunking. The maximum size of a Transfer-Encoding: chunked response body and a Transfer-Encoding: chunked request body can be set to determine how much data is buffered before a network write is performed.

Procedure

1. Click **Servers>Proxy Servers>***proxy_server_name*.
2. Under Proxy Settings, expand **HTTP Proxy Server Settings** and click **Denial of service protection**
3. Set the appropriate buffer size in kilobytes for **Maximum request body buffer size**. Buffering can lead to better performance because it will decrease the number of network writes for large payloads. The size of the buffer must not be configured too high or memory exhaustion can occur. To determine the optimal values for your environment, gradually increase the size of the buffer until the proper balance has been achieved.
4. Set the appropriate buffer size in kilobytes for **Maximum response body buffer size**. The same precaution must be taken when setting Maximum response body buffer size as were indicated for Maximum request body buffer size.

Results

After these steps have been properly complete, denial of service protection will be in place for your proxy server or DMZ Secure Proxy Server for IBM WebSphere Application Server.

Denial of Service Protection

Use these settings to define denial of service protection for the proxy server. The buffer chunk sizes are used to provide protection against denial of service attacks. It is very important to tune these settings to balance the level of protection with the performance impacts that can be experienced.

To view this administrative console page, click **Servers** > **Proxy Servers** > **<server_name>** > **HTTP Proxy Server Settings** > **Denial of Service Protection**.

Maximum request body buffer chunk size

This field sets the maximum amount of the request body data that is buffered in memory before it is sent. The proxy server and secured proxy server can buffer multiple network reads of a Transfer-Encoding: chunked request body before performing a network write. This can improve performance because buffering the data will decrease the number of network writes required for large payloads. This field should be adjusted with caution. If the buffer size is set too high, memory exhaustion, which is a common intent of denial of service attacks, might be experienced. You will need to test different buffer sizes to find the optimal level of balance between protection and performance for your system. The default value of this field is 32 kilobytes.

Maximum response body buffer chunk size

This field sets the maximum amount of the response body data that is buffered in memory before it is sent. The proxy server and secured proxy server can buffer multiple network reads of a Transfer-Encoding: chunked response body before performing a network write. This can improve performance because buffering the data will decrease the number of network writes required for large payloads. This field should be adjusted with caution. If the buffer size is set too high, memory exhaustion, which is a common intent of denial of service attacks, might be experienced. You will need to test different buffer sizes to find the optimal level of balance between protection and performance for your system. The default value of this field is 32 kilobytes.

Tuning the security properties for the DMZ Secure Proxy Server for IBM WebSphere Application Server

When creating a DMZ Secure Proxy Server for IBM WebSphere Application Server, default security levels of high, medium and low are available. In addition to the predefined configuration levels, you can modify the security settings for your DMZ Secure Proxy Server for IBM WebSphere Application Server. When you choose to customize the settings, a qualitative value of high, medium or low is still assigned to inform you of the overall security level of your DMZ Secure Proxy Server for IBM WebSphere Application Server.

Before you begin

A DMZ Secure Proxy Server for IBM WebSphere Application Server must be installed before these steps can be completed. The DMZ Secure Proxy Server for IBM WebSphere Application Server profile must be registered with the AdminAgent for this panel to be available.

About this task

Installing the DMZ Secure Proxy Server for IBM WebSphere Application Server in the DMZ rather than the secured zone presents new security challenges. The DMZ Secure Proxy Server for IBM WebSphere Application Server has been equipped with various capabilities to provide protection for meeting these challenges. It addition to the predefined security configurations for your DMZ Secure Proxy Server for IBM WebSphere Application Server you can also tune the settings to customize the protection.

Procedure

1. Click **Servers** > **Proxy Servers** > **<secured_proxy_server_name>** > **Custom security settings** to open up the Proxy security settings panel.
2. Choose your administration option for your DMZ Secure Proxy Server for IBM WebSphere Application Server.
   - Local Administration - Security level: medium and high.

     This option allows two different types of administration. Managing the DMZ Secure Proxy Server for IBM WebSphere Application Server entirely using the wsadmin tool and loading updated profiles imported from inside the cell using the wsadmin tool are both considered Local Administration.
   - Remote Administration- Security level: low
3. Choose your routing option for your DMZ Secure Proxy Server for IBM WebSphere Application Server.
   - Static routing - Security level: high
   - Dynamic routing - Security level: low and medium
4. Choose your startup permission option for your DMZ Secure Proxy Server for IBM WebSphere Application Server.
   - Run as an unprivileged user - Security level: medium and high
   - Run as privileged user - Security level: low
5. Optional: If Run as an unprivileged user is selected, enter the user name or the user group whose identity the server should assume after startup has completed.
6. Choose your custom error page policy option for your DMZ Secure Proxy Server for IBM WebSphere Application Server.
   - Local error page handling - Security level: high

     If you choose to use local error page handling, you need to select which error responses should use custom error messages. Select Handle local errors for responses generated by the proxy server and select Handle remote errors for responses generated by the backend server. Both options may be selected to use custom error messages for local and remote errors. Manage your error code mappings to determine the custom error pages to be used for specific responses.
   - Remote error page handling - Security level: low and medium

If you choose to use remote error page handling to include custom errors, you need to select which error responses should be customized. Select Handle local errors for responses generated by the proxy server and select Handle remote errors for responses generated by the backend server. Both options may be selected to use custom error messages for both local and remote errors. Manage the headers that should be sent to the custom error application and what status codes are to be recognized as errors.

Results

You have finished customizing the security settings for your DMZ Secure Proxy Server for IBM WebSphere Application Server. A qualitative value of high, medium or low has been calculated based on the settings you have chosen to demonstrate the Current DMZ Security level.

DMZ Secure Proxy Server for IBM WebSphere Application Server start up user permissions

The overall security level of the DMZ Secure Proxy Server for IBM WebSphere Application Server can be hardened by reverting the server process to run as an unprivileged user after startup. Although the DMZ Secure Proxy Server for IBM WebSphere Application Server must be started as a privileged user, changing the server process to run as an unprivileged user provides additional protection for local operating resources.

Like the proxy server, the DMZ Secure Proxy Server for IBM WebSphere Application Server must start under a privileged user because it requires authorization to initialize privileged ports. Ports lower than 1024 are considered privileged ports. After these ports are initialized and access to the protected ports is no longer required, it is possible to change the user association of the DMZ Secure Proxy Server for IBM WebSphere Application Server process. Altering the server process to run using the privileges of a user or a group that does not have authority to access the local operation system resources adds a layer of protection to those resources. The firewall helps protect local operating system resources for the proxy server, but as the DMZ Secure Proxy Server for IBM WebSphere Application Server is installed in the DMZ, this type of protection becomes a higher priority. Although changing the user association of the server process for the DMZ Secure Proxy Server for IBM WebSphere Application Server is not required, continuing to run as a privileged user does not use the extra layer of protection for local operation resources that is provided when the server process is changed to run as an unprivileged user.

Table 31. Start up options. This table describes the proxy server start up options.

Run as unprivileged user	This is considered a high and medium security level setting.
Run as privileged user	This is considered a low security level setting.

DMZ Secure Proxy Server for IBM WebSphere Application Server routing considerations

This topic summarizes some of the security implications that must be considered when choosing how your DMZ Secure Proxy Server for IBM WebSphere Application Server will match incoming HTTP requests to an application or routing rule.

The DMZ Secure Proxy Server for IBM WebSphere Application Server can be configured to route requests either statically or dynamically. Using static routing specifies routing is performed using a flat configuration file using routing precedence that is inherent to the ordering of the directives. Requests can also be routed dynamically using a best match mechanism that determines the installed application or routing rule that corresponds to a specific request. The DMZ Secure Proxy Server for IBM WebSphere Application Server will dynamically discover the best route to a destination and distribute to servers with like protocols. It is considered more secure to use static routing than dynamic routing. This is because dynamic routing requires the secured proxy server to communicate with the application server to map requests, whereas static routing is self-contained by the secure proxy server and requires no additional communication.

Table 32. Routing options. This table lists the proxy routing options.

Static Routing	This is considered a high security level setting.
Dynamic Routing	This is considered a low and medium security level setting.

DMZ Secure Proxy Server for IBM WebSphere Application Server administration options

The DMZ Secure Proxy Server for IBM WebSphere Application Server is administered differently than the WebSphere proxy server. The DMZ Secure Proxy Server for IBM WebSphere Application Server is a separate binary installed in the DMZ. Installing the DMZ Secure Proxy Server for IBM WebSphere Application Server in the DMZ requires that administration be managed differently for security reasons. Several administrative options are available for administering the DMZ Secure Proxy Server for IBM WebSphere Application Server to provide different levels of balance between security and usability.

The most secure way to administer the DMZ Secure Proxy Server for IBM WebSphere Application Server is locally using the wsadmin tool. The DMZ Secure Proxy Server for IBM WebSphere Application Server does not have web container. Therefore, local administration can only be done via the command line. Using the wsadmin commands locally to manage the DMZ Secure Proxy Server for IBM WebSphere Application Server is the most secure option available because it does not require any external listening ports to be opened.

The DMZ Secure Proxy Server for IBM WebSphere Application Server configurations can also be managed within the network deployment application server cell and then imported locally using the wsadmin commands. The configurations are maintained inside the cell as configuration only profiles. The profiles are registered with the Admin Agent and are then managed using the administrative console. After you implement any changes to the profile, you export the configuration to a configuration archive (CAR) file using the exportProxyProfile or exportProxyServer wsadmin commands. After you transmit the CAR file to the local DMZ Secure Proxy Server for IBM WebSphere Application Server installation using ftp, the CAR file is imported using the importProxyProfile or importProxyServer wsadmin commands. This option is also considered to be local administration.

Note: Due to security reasons, the number of listening ports on the secure proxy is minimized. You might not be able to manage, start, stop the secure proxy from the admin agent or the job manager remotely when admin security is enabled.

Error handling security considerations for the DMZ Secure Proxy Server for IBM WebSphere Application Server

The overall security level of the DMZ Secure Proxy Server for IBM WebSphere Application Server is partially determined by the choices made regarding the handling of custom errors.

You can define a custom error page for each error code or a group of error codes on errors generated by the proxy server or the application server. This is done using HTTP status codes in responses to generate uniform customized error pages for the application. For security reasons, you can ensure that the error pages are read from the local file system instead of being forwarded to a custom remote application. Choosing this option limits the code path and eliminates the need for a potentially unauthorized application to be run as the error message is generated based on a flat file. For more information about the error handling for the secured proxy server, see “Overview of the custom error page policy” on page 287.

The following security level settings are used when evaluating a custom security level. Local error page handling is used for all of the predefined security levels.

Table 33. Error handling options. This table lists the security level settings that are used when evaluating a custom security level.

Local error page handling	This is considered a high security level setting.

Table 33. Error handling options (continued). This table lists the security level settings that are used when evaluating a custom security level.

Remote error page handling	This is considered a medium and low security level setting.

Proxy security level properties

These settings describe the attributes and policies that define the security level of a secured proxy server. The overall security level of the secured proxy server is set to the weakest level of security assigned to any of the individual settings.

To view this administrative console page, click **Servers > Proxy Servers >** *server_name* **> Custom security settings**. This panel will only be available for a secure proxy server profile that has been registered with the AdminAgent.

Current security level

A qualitative security level based on an evaluation of the current security related configuration values.

The possible values for Current DMZ Security are high, medium, low. During creation of the secured proxy server, default configurations of high, medium and low are available. You are also able to customize these security settings resulting in the Current DMZ Security level being calculated by the system. Each custom setting has an assigned value of high, medium or low. The overall security level is equal to value of the setting that is considered the least secure. For example, to have an overall security level of high, all settings must be configured to the values associated with a high level of security. If any of the settings are configured with a less secure value, the overall security level is the value of that setting.

Administration

Table 34. Administration options. This table lists the proxy administration options.

Option	Used as the default value in the predefined security levels	Description
Local administration	The default value for the Medium and the High security levels	Specifies that administration of the secure proxy server can only be performed using wsadmin commands performed locally on the system.
Remote administration	The default value for the Low security level	Specifies that remote administration of the secure proxy server is permitted.

Routing

Table 35. Routing options. This table lists the proxy routing options.

Option	Used as the default value in the predefined security levels	Description
Static routing	The default value for the High security level	Specifies that the proxy server will make routing determinations from routing information based on flat files on the file system. This is for Hypertext Transfer Protocol (HTTP) only
Dynamic routing	The default value for the Low and the Medium security levels	Specifies that the proxy server will dynamically discover the best route to a destination and distribute to servers with like protocols.

Start-up permissions

Table 36. Start-up permission options. This table lists the proxy start-up permission options.

Option	Used as the default value in the predefined security levels	Description
Run as an unprivileged user	The default value for the Medium and the High security levels	Specifies that the server process will revert to a predefined unprivileged user after start-up has completed.

Table 36. Start-up permission options (continued). This table lists the proxy start-up permission options.

Option	Used as the default value in the predefined security levels	Description
Run as a privileged user	The default value for the Low security level	Specifies that the server process does not revert to an unprivileged user after startup. It is a requirement that the proxy server start under a privileged user as it initializes privileged ports. Ports lower than 1024 are considered privileged ports. Under this setting, the effective user of the server process continues to be the privileged user. This setting does not provide additional hardening to the access of the server process to the local operation system resources. This is considered a low security level setting.

Custom Error Page Policy

Table 37. Error page options. This table lists the proxy error page options.

Option	Used as the default value in the predefined security levels	Description
Local error page handling	The default value for the Low, the Medium and the High security levels	Specifies that error responses will be generated from flat custom error page files stored locally on the local file system.
Remote error page handling	None	Specifies to route error responses to a remote custom application deployed on a back-end server. This application will generate a custom response for the error

Local error page handling

- **Handle errors generated by the proxy server**

  Specifies if errors generated by the proxy server should be handled with the custom static error pages stored on the local file system. If this is not selected then the default error messages will be used instead of any custom error pages.
- **Handle errors generated by application servers**

  Specifies if errors generated by the backend server should be handled with the custom static error pages stored on the local file system. If this is not selected then the default error messages will be used instead of any custom error pages.
- **Error mappings**

  Specifies the error codes to match with specific static error pages stored on the file system. You can use a relative file path under the configured static file document root to assign a custom error file to be used for a specific error code or group of error codes. The wildcard character, * , is used to assign error files to groups of error codes.

Remote error page handling

- **Error page generation application URI**

  Specifies the URI for the custom error page generation application.
- **Handle errors generated by the proxy server**

  Specifies if errors generated by the proxy server should be handled with the custom error application deployed on the application server. If this is not selected then the default error messages will be used instead of any custom error pages.
- **Handle errors generated by application servers**

  Specifies if errors generated by the backend server should be handled with the custom error application deployed on the application server. If this is not selected then the default error messages will be used instead of any custom error pages.
- **Headers to forward to Error page Application**

Specifies a list of the headers from the original request to forward to the error page generation application.

- **HTTP status codes that are to be recognized as errors**

  Specifies a list of the status codes in a response that should be directed to the error page generation application.

Configuring a DMZ Secure Proxy Server for IBM WebSphere Application Server using the administrative console

You can create a DMZ Secure Proxy Server for IBM WebSphere Application Server inside of a network deployment cell using the administrative console of an administrative agent. You can then export the secure proxy server to a node in the demilitarized zone (DMZ) into which you can then import the configuration. After the secure proxy server is created on a node in the DMZ, administration can be done locally or it can be done using the Job Manager console.

Before you begin

Before you begin, complete these tasks.

1. Review the content of the topic "Choosing a front end for your WebSphere Application Server topology". This topic helps you determine whether you should set up a web server plug-in, a proxy server, or a secure proxy server to provide session affinity, failover support, and workload balancing for your WebSphere Application Server topology.
2. Install the DMZ Secure Proxy Server for IBM WebSphere Application Server code.

   See the Installing the DMZ Secure Proxy Server for IBM WebSphere Application Server image topic for more details.
3. Create a secure proxy server (configuration-only) profile on a network-deployment installation using either the Profile Management Tool or the manageprofiles command.

   See the page about creating secure proxy profiles for more information on creating the profile using the Profile Management Tool.
4. Create an administrative agent profile on the network-deployment installation using either the Profile Management Tool or the manageprofiles command.

   See the page about creating management profiles with administrative agents for additional details.

About this task

The DMZ Secure Proxy Server for IBM WebSphere Application Server does not contain a web container and therefore does not have an administrative console. Secure proxy server configurations can also be managed within a network deployment application server cell and then imported locally into the DMZ Secure Proxy Server for IBM WebSphere Application Server using wsadmin commands. The configurations are created and maintained inside the network deployment application server cell as configuration-only profiles. The profiles are registered with the administrative agent and are then managed using the administrative console. You configure the secure proxy server profile in the network deployment application server cell, export the configuration to a configuration archive (CAR) file using the exportProxyProfile or exportProxyServer wsadmin command, transmit the CAR file to the local secure proxy server installation using FTP, and import the configuration into the DMZ Secure Proxy Server for IBM WebSphere Application Server using the importProxyProfile or importProxyServer wsadmin command. You then repeat the process if any changes are made to the secure proxy server configuration.

Procedure

1. Start an administrative agent on a network-deployment installation.
2. Register the secure proxy (configuration-only) profile with the administrative agent using the registerNode command.

3. Restart the administrative agent.
4. When the administrative agent prompts you with a list of the nodes that it manges, select the node from the secure proxy (configuration-only) profile.
5. From the administrative console of the administrative agent, select **Servers > Server Types > WebSphere proxy servers**.
6. Click **New** to access the Proxy Server Creation wizard.
7. Select the DMZ Secure Proxy Server for IBM WebSphere Application Server node.
8. Complete the steps in the wizard to create a new secure proxy server.

   This new secure proxy server will only be used as a configuration. It cannot be started inside of the cell.
9. Set the **sipClusterCellName** custom property to be the cell name that contains the configured cluster of SIP containers. This step applies to the SIP proxy only, and not to the HTTP server. For more information about how to set this custom property, see the topic "SIP proxy server custom properties" in this information center.
10. Save the configuration.
11. Using the wsadmin tool, connect to the secure proxy server profile.
12. Export your configuration to be used inside of the DMZ; you can export the entire profile or export the server.
    - To export the profile, run the exportProxyProfile command. For example:
      ```
      AdminTask.exportProxyProfile('[-archive "c:/myCell.car"]')
      ```
    - To export the server, run the exportProxyServer command. For example:
      ```
      AdminTask.exportProxyServer ('[-archive c:\myServer.ear -nodeName node1 -serverName proxy1]')
      ```
13. Using FTP, transfer the configuration archive to the local secure proxy server node.
14. Start the wsadmin tool on the secure proxy server profile.
15. Import the entire profile, or import the server.
    - Use the importProxyProfile command to import the profile. In the following example, the existing secure proxy server in the profile is replaced with the server in the imported proxy profile; for example:

      Windows
      ```
      AdminTask.importProxyProfile(['-archive', 'c\myCell.car', '-deleteExistingServers', 'true'])
      ```
      Linux Solaris AIX HP-UX
      ```
      AdminTask.importProxyProfile(['-archive', '/myCell.car', '-deleteExistingServers', 'true'])
      ```
    - Use the importProxyServer command to import the server. In the following example, the existing secure proxy server is replaced with the imported proxy server; for example:

      Windows
      ```
      AdminTask.importProxyServer('[-archive c:\myServer.ear -nodeInArchive node1 -serverInArchive proxy1 -deleteExistingS
      ```
      Linux Solaris AIX HP-UX
      ```
      AdminTask.importProxyServer('[-archive /myServer.ear -nodeInArchive node1 -serverInArchive proxy1 -deleteExistingSer
      ```
16. Save the configuration changes.

    Use the following command example to save your configuration changes:
    ```
    AdminConfig.save()
    ```

Results

Successful completion of this procedure results in deployment of the DMZ Secure Proxy Server for IBM WebSphere Application Server on a node in the DMZ.

What to do next

You can now start and begin to use your DMZ Secure Proxy Server for IBM WebSphere Application Server.

Configure secure routing for a DMZ Secure Proxy Server for IBM WebSphere Application Server

You can configure the DMZ Secure Proxy Server for IBM WebSphere Application Server to route requests statically or dynamically.

Before you begin

Configure your profiles and security properties before you configure routing. See the topic Tuning the security properties for the DMZ Secure Proxy Server for IBM WebSphere Application Server. Decide whether you want to configure static or dynamic routing.

About this task

Static routing is performed using a flat configuration file. Static routing is considered more secure than dynamic routing. With dynamic routing, requests are routed through a best match mechanism that determines the installed application or routing rule that corresponds to a specific request. The secure proxy server will dynamically discover the best route to a destination and distribute to servers with like protocols.

The secure routing options are:

- Use static routing with the exportTargetTree command.
- Use dynamic routing by setting up a core group bridge tunnel. See the topic Configuring communication with a core group that resides on a DMZ Secure Proxy Server for IBM WebSphere Application Server.

gotcha: Because the DMZ secure proxy server resides in a different cell from the application servers, it must be configured to trust the application server cell in order for Secure Sockets Layer (SSL) to work properly. See the third step in this procedure.

Use the following procedure to configure static or dynamic secure routing.

Procedure

- To configure static routing, follow these steps:
  1. Set the secure proxy server to use static routing, which is the default level after installation. You can do this by either setting the overall security level to `high` or by setting the custom security level for the routing property to `static`.
  2. Use the wsadmin tool to query for the TargetTreeMbean mbean.
     ```
     mbean=AdminControl.queryNames('*:*,type=TargetTreeMbean,process=dmgr')
     ```
  3. If your application uses Servlet 3.0 dynamic cookies, start the application that uses dynamic cookies.
  4. Invoke the exportTargetTree method on the TargetTree mbean to a specified XML file.
     ```
     AdminControl.invoke(mbean, 'exportTargetTree', '/opt/IBM/WebSphere/AppServer/targetTree.xml')
     ```
     The static routing file is a special type of routing file that the proxy server uses to route a request from the proxy server directly to an application server. It is not used to route requests from the Web server plug-in to an application server.
  5. Using the deployment manager command line, transfer the `targetTree.xml` file from the deployment manager to the *profile_root*/`staticRoutes` directory for the proxy server.

The file is transferred from the deployment manager to the proxy server by FTP or some other protocol.

If your secure proxy server is interfacing with multiple cells, you can add the static routing file for each cell to the *profile_root*/staticRoutes directory. The secure proxy server considers any *xxx*.xml file that is included in the in the *profile_root*/staticRoutes directory a static routing file. The file does not have to be named targetTree.xml.

When you add multiple *xxx*.xml files to the *profile_root*/staticRoutes directory for a secure proxy server, the secure proxy server merges the content of all of these files. The result of this merge is a single static routing file that the secure proxy server can use to route requests to servers in any of the cells.

gotcha: The merger process does not include any files in subdirectories of the *profile_root*/staticRoutes directory. Therefore the secure proxy server does not use any content that is contained in files located in any subdirectory when it routes requests to a servers.

Any attribute change that affects the content of the static routing file, such as a virtual host change, addition or deletion of a proxy server cluster member, a change in weight of a proxy server cluster member, or the installation or uninstallation of an application, automatically regenerates the static routing file.

6. Start the proxy server from the system command line:

```
profile_root/startServer proxy_server_name
```

- To configure dynamic routing, follow these steps:
  1. Configure the core group bridge in the application server cell. See the topic Configuring communication with a core group that resides on a DMZ Secure Proxy Server for IBM WebSphere Application Server.
  2. Export the tunnel template settings to a file. From the wsadmin tool, use the exportTunnelTemplate command to export the settings, as in the following example:

```
AdminTask.exportTunnelTemplate('[-tunnelTemplateName exportedTunnelTemplate
-outputFileName tunnelTemplate1.props]')
```

  3. Import the tunnel template settings into the DMZ proxy configuration, as in the following example:

```
AdminTask.importTunnelTemplate('[-inputFileName tunnelTemplate1.props
-bridgeInterfaceNodeName DMZNode01 -bridge InterfaceServerName DMZProxyServer01]')
```

  4. Start the proxy server from the system command line:

```
profile_root/startServer proxy_server_name
```

- To configure SSL communications, follow these steps:
  1. Configure the ssl.client.props properties file using the retrieveSigners command. See the information center topic on using the retrieveSigners command for more details.
  2. The com.ibm.ssl.trustStore property should be set to point to the secure proxy server trust.p12 file. For example:

```
${user.root}/config/cells/SecureProxyCell1/nodes/SecureProxyNode1/trust.p12
```

  3. Specify the truststore name of the cell in which the application servers reside when running the command. By default, its name is CellDefaultTrustStore. The retrieveSigners command can then be used to update the secure proxy server to trust the application server cell:

```
retrieveSigners CellDefaultTrustStore AnotherTrustStore -host mybackendDmgr.location.com -port 8879
```

Results

Completing this procedure results in configuring secure routing for a DMZ Secure Proxy Server for IBM WebSphere Application Server.

What to do next

You can now start and begin to use the DMZ Secure Proxy Server for IBM WebSphere Application Server.

WebSphere DMZ Secure Proxy Server for IBM WebSphere Application Server

The DMZ Secure Proxy Server for IBM WebSphere Application Server can be used to provide a secure platform for your proxy server.

The DMZ Secure Proxy Server for IBM WebSphere Application Server installation enables you to install your proxy server in the demilitarized zone (DMZ), while reducing the security risk that might occur if you choose to install an application server in the DMZ to host a proxy server. The risk is reduced by removing any functionality from the application server that is not required to host the proxy servers, but that can pose a security risk. Installing the secure proxy server in the DMZ rather than the secured zone presents new security challenges. However, the secure proxy server is equipped with capabilities to provide protection from these challenges.

The following capabilities are available to harden the security of the DMZ Secure Proxy Server for IBM WebSphere Application Server and to determine the level of security to assign.

- Startup user permissions

  The secure proxy server process can be changed to run as an unprivileged user after startup. Although the secure proxy server must be started as a privileged user, changing the server process to run as an unprivileged user provides additional protection for local operating resources.
- Routing considerations

  The secure proxy server can be configured to route requests to target servers based on static routing information or dynamic information. Static routing means that the server obtains the routing information from local flat files. Dynamic routing means that the server obtains the routing information from a Hypertext Transfer Protocol (HTTP) tunnel connection from the proxy server to a server in the secure zone. It is more secure to use static routing as the use of dynamic routing requires an additional connection through the inner firewall. Static routing is only applicable to HTTP requests.
- Administration options

  The secure proxy server does not contain a web container, and therefore is unable to host the administrative console. It is better to not have a web container on a DMZ Secure Proxy Server for IBM WebSphere Application Server since hosting application artifacts is considered a security risk and adds an unnecessary footprint to the proxy server. The secure proxy server is installed separately and has several different administrative options that have security implications.
- Error handling

  Custom error pages can be used by the secure proxy server for specific error codes or groups of error codes. A custom error page application can be used to generate error messages or flat custom error page files can be stored locally on the file system and used during run time. Choosing to use flat custom error pages instead of a custom error application provides a higher level of security. Choosing this option limits the code path and eliminates the need for a potentially unauthorized application to be run when an error page is needed.
- Denial of service protection

  Denial of service protection is provided with the inclusion of two properties: Maximum request body buffer chunk size and Maximum response body buffer chunk size. These properties must be tuned to balance the level of protection with the performance overhead that might be experienced if these properties are set incorrectly.

When creating the DMZ Secure Proxy Server for IBM WebSphere Application Server, you can choose any of the default security levels: High, Medium or Low.

- Low DMZ security level

Table 38. Low DMZ security level default values. This table describes the settings and default values for the low DMZ security level.

Setting	Default value
Startup permissions	Run as a privileged user
Routing	Dynamic routing
Administration	Remote Administration
Error handling	Local error page handling

- Medium DMZ security level

Table 39. Medium DMZ security level default values. This table describes the settings and default values for the medium DMZ security level.

Setting	Default value
Startup permissions	Run as an unprivileged user
Routing	Dynamic routing
Administration	Local Administration
Error handling	Local error page handling

- High DMZ security level

Table 40. High DMZ security level default values. This table describes the settings and default values for the high DMZ security level.

Setting	Default value
Startup permissions	Run as an unprivileged user
Routing	Static routing
Administration	Local Administration
Error handling	Local error page handling

Important: The High DMZ security level cannot be used for SIP proxy servers, because static routing cannot be used for the SIP proxy server.

In addition to these predefined settings, you can customize the settings to better serve your requirements. If you choose to customize the settings, your DMZ Secure Proxy Server for IBM WebSphere Application Server is assigned a qualitative categorization of your security level called the *current* security level. Each custom setting has been assigned a value of High, Medium or Low. The current security level is equal to the value of the least secure setting being used. To achieve a current security level of High, only settings assigned the high value can be configured. To achieve a current security level of Medium, only settings with values of High or Medium can be used. A current security level of Low is used if any settings that are assigned the value of Low are set.

An additional change to enhance the protection for the DMZ Secure Proxy Server for IBM WebSphere Application Server is the switch from a Java Development Kit (JDK) to a Java Runtime Environment (JRE). Switching from a JDK to a JRE removes the inclusion of a compiler on the installation. This change is beneficial because the compiler can possibly be used for malicious purposes in the event of a security breach.

Appendix. Directory conventions

References in product information to *app_server_root*, *profile_root*, and other directories imply specific default directory locations. This topic describes the conventions in use for WebSphere Application Server.

Default product locations (distributed)

The following file paths are default locations. You can install the product and other components or create profiles in any directory where you have write access. Multiple installations of WebSphere Application Server Network Deployment products or components require multiple locations. Default values for installation actions by root and nonroot users are given. If no nonroot values are specified, then the default directory values are applicable to both root and nonroot users.

app_client_root

Table 41. Default installation root directories for the Application Client for IBM WebSphere Application Server.

This table shows the default installation root directories for the Application Client for IBM WebSphere Application Server.

User	Directory
Root	AIX `/usr/IBM/WebSphere/AppClient` (Java EE Application client only) HP-UX Linux Solaris `/opt/IBM/WebSphere/AppClient` (Java EE Application client only) Windows `C:\Program Files\IBM\WebSphere\AppClient`
Nonroot	AIX HP-UX Linux Solaris `user_home/IBM/WebSphere/AppClient` (Java EE Application client only) Windows `C:\IBM\WebSphere\AppClient`

app_server_root

Table 42. Default installation directories for WebSphere Application Server.

This table shows the default installation directories for WebSphere Application Server Network Deployment.

User	Directory
Root	AIX `/usr/IBM/WebSphere/AppServer` HP-UX Linux Solaris `/opt/IBM/WebSphere/AppServer` Windows `C:\Program Files\IBM\WebSphere\AppServer`
Nonroot	AIX HP-UX Linux Solaris `user_home/IBM/WebSphere/AppServer` Windows `user_home\IBM\WebSphere\AppServer`

component_root
The component installation root directory is any installation root directory described in this topic. Some programs are for use across multiple components—in particular, the Web Server Plug-ins, the Application Client, and the IBM HTTP Server. All of these components are part of the product package.

gskit_root
IBM Global Security Kit (GSKit) can now be installed by any user. GSKit is installed locally inside

the installing product's directory structure and is no longer installed in a global location on the target system. The following list shows the default installation root directory for Version 8 of the GSKit, where *product_root* is the root directory of the product that is installing GSKit, for example IBM HTTP Server or the web server plug-in.

AIX HP-UX Linux Solaris

product_root/gsk8

Windows

product_root\gsk8

profile_root

Table 43. Default profile directories.

This table shows the default directories for a profile named profile_name on each distributed operating system.

User	Directory
Root	AIX /usr/IBM/WebSphere/AppServer/profiles/*profile_name* HP-UX Linux Solaris /opt/IBM/WebSphere/AppServer/profiles/*profile_name* Windows C:\Program Files\IBM\WebSphere\AppServer\profiles\profile_name
Nonroot	AIX HP-UX Linux Solaris *user_home*/IBM/WebSphere/AppServer/profiles Windows *user_home*\IBM\WebSphere\AppServer\profiles

plugins_root

Table 44. Default installation root directories for the Web Server Plug-ins.

This table shows the default installation root directories for the Web Server Plug-ins for WebSphere Application Server.

User	Directory
Root	AIX /usr/IBM/WebSphere/Plugins HP-UX Linux Solaris /opt/IBM/WebSphere/Plugins Windows C:\Program Files\IBM\WebSphere\Plugins
Nonroot	AIX HP-UX Linux Solaris *user_home*/IBM/WebSphere/Plugins Windows C:\IBM\WebSphere\Plugins

wct_root

Table 45. Default installation root directories for the WebSphere Customization Toolbox.

This table shows the default installation root directories for the WebSphere Customization Toolbox.

User	Directory
Root	AIX /usr/IBM/WebSphere/Toolbox HP-UX Linux Solaris /opt/IBM/WebSphere/Toolbox Windows C:\Program Files\IBM\WebSphere\Toolbox

Table 45. Default installation root directories for the WebSphere Customization Toolbox (continued).

This table shows the default installation root directories for the WebSphere Customization Toolbox.

User	Directory
Nonroot	AIX HP-UX Linux Solaris *user_home*/IBM/WebSphere/Toolbox Windows C:\IBM\WebSphere\Toolbox

web_server_root

Table 46. Default installation root directories for the IBM HTTP Server.

This table shows the default installation root directories for the IBM HTTP Server.

User	Directory
Root	AIX /usr/IBM/HTTPServer HP-UX Linux Solaris /opt/IBM/HTTPServer Windows C:\Program Files\IBM\HTTPServer
Nonroot	AIX HP-UX Linux Solaris *user_home*/IBM/HTTPServer Windows C:\IBM\HTTPServer

Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, is the user's responsibility.

APACHE INFORMATION. This information may include all or portions of information which IBM obtained under the terms and conditions of the Apache License Version 2.0, January 2004. The information may also consist of voluntary contributions made by many individuals to the Apache Software Foundation. For more information on the Apache Software Foundation, please see http://www.apache.org. You may obtain a copy of the Apache License at http://www.apache.org/licenses/LICENSE-2.0.

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

IBM Director of Intellectual Property & Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
USA

Trademarks and service marks

Index