WebSphere® Application Server for z/OS, Version 6.1

Setting up the application serving environment

GA22-7958-04

Note

Before using this information, be sure to read the general information under “Notices” on page 577.

Compilation date: May 5, 2006

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 919-254-0206.

  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.

Chapter 1. Configuring the product after installation

Use this task to configure WebSphere Application Server for z/OS application serving environments for your z/OS target systems.

Configuring a WebSphere Application Server for z/OS application serving environment consists of setting up the WebSphere Application Server for z/OS configuration directory for the environment, making any required changes to the z/OS target system that pertain to the particular application serving environment, and starting the new environment to verify the configuration. Configuring these application serving environments after product installation requires a fair amount of planning and coordination. If you have not previously configured WebSphere Application Server for z/OS, you should configure a ″practice″ stand-alone application server using the default options then proceed to configure the actual product configuration that you want. See Building a practice WebSphere Application Server for z/OS cell for more information.

WebSphere Application Server for z/OS application serving environment nodes can be created using either the ISPF-based Customization Dialog (see “Configuring with the ISPF Customization Dialog” on page 2) or the workstation-based Profile Management tool (see “Configuring with the Profile Management tool” on page 93).

Once a node is configured and running, make further changes using the Web-based administrative console or scripting.

Once your application serving environment is up and running, you can install and test applications.

Downloading archive images from the z/OS platform to a distributed platform

Follow these steps to set up either JDBC or WebSphere Application Clients on a distributed platform workstation.

The product tape for the WebSphere Application Server product that runs on z/OS includes image archives for the following functions that can be run on a distributed platform:

- WebSphere Application Clients for distributed platforms supported by the WebSphere Application Server Version 6 products.
- The Data Direct Java Database Connectivity (JDBC) drivers for Microsoft Windows platforms.

1. Create a directory on the workstation or server where you want to install the image archive files for WebSphere Application Client or Data Direct Java Database Connectivity (JDBC) drivers for Microsoft Windows platforms.
2. For example, on the z/OS HFS containing the WebSphere Application Server product, locate the WebSphere Application Client image archive files.

   The image archives for each supported platform are contained in the following directory on the HFS.

   ```
   usr/lpp/zWebSphere/V6R0/downloads2...
   .../platform
        /zdownloads.inst.cd.image.part1.for_platform.archive-ending(s))
        /zdownloads.inst.cd.image.part2.for_platform.archive-ending(s))
   ```

   There are two install image archives for each supported platform. Each pair of install images archives includes that includes the WebSphere Application Client archive files for that platform. The install image archives also contain the Data Direct Java Database Connectivity (JDBC) drivers for Microsoft Windows platforms.

   Both install image archives must be downloaded to the same directory on the distributed platform.

3. Use FTP or some other process to download the two image archives to the distributed platform. These files are in binary format and must be downloaded in that format.
4. Archive and expand both of the files into the directory you created in Step 1.

See Installing Application Client for WebSphere Application Server for a description of how to install the Application Client on the distributed platform.

Configuring with the ISPF Customization Dialog

Use this task to configure WebSphere Application Server for z/OS application serving environments for your z/OS target systems.

- Choose a z/OS target system and complete the steps in the ″Installing the product and additional software″ and ″Preparing the base operating system″ sections of the *Installing your application serving environment* PDF.
- Choose a WebSphere Application Server for z/OS configuration (practice, stand-alone or Network Deployment cell) and complete the steps in the ″Planning for product configuration″ section of the *Installing your application serving environment* PDF.

Note: The ISPF Customization Dialog is deprecated; it will be removed in a future release.

Configuring a WebSphere Application Server for z/OS application serving environment consists of setting up the WebSphere Application Server for z/OS configuration directory for the environment, making any required changes to the z/OS target system that pertain to the particular application serving environment, and starting the new environment to verify the configuration. Configuring these application serving environments after product installation requires a fair amount of planning and coordination. If you have not previously configured WebSphere Application Server for z/OS, you should configure a ″practice″ stand-alone application server using the default options then proceed to configure the actual product configuration that you want.

If you have already created a Network Deployment cell, follow the instructions in this section to expand the cell by creating a new managed node or federating an existing stand-alone application server node into the Network Deployment cell.

WebSphere Application Server for z/OS application serving environment nodes are created using the ISPF-based Customization Dialog. Once a node is configured and running, make further changes using the Web-based administrative console or scripting.

After you have installed the WebSphere Application Server for z/OS product, prepared your z/OS target systems, and planned your new application serving environment, perform the tasks in this section to configure and start the application serving environment.

1. Review the use of the Customization Dialog. See “Using the Customization Dialog” on page 3 for more information.
2. If this application serving environment uses a new security domain, create the security domain before proceeding. See “Creating common MVS groups and users” on page 8 for detailed instructions.
3. Follow the directions for the type of application serving environment you want to configure:
   - “Creating a stand-alone application server cell” on page 19
   - “Creating a Network Deployment cell” on page 46
   - “Creating a managed server node” on page 66
   - “Federating a stand-alone application server into a Network Deployment cell” on page 85

Once your application serving environment is up and running, you can install and test applications. You might also want to configure your Web servers to interact with WebSphere Application Server for z/OS.

See the Related Tasks section for additional tasks you can perform once your application serving environment is configured.

Using the Customization Dialog

This article provides general information on starting and using the Customization Dialog. See the instructions for each customization task for detailed directions on using the Customization Dialog to perform that particular task.

The WebSphere Application Server for z/OS Customization Dialog is an ISPF dialog, running under TSO, that you use for the initial setup of WebSphere Application Server for z/OS cells and nodes. The Customization Dialog itself does not create the cells and nodes. Instead, it creates batch jobs, scripts, and data files that you then use to perform WebSphere Application Server for z/OS customization tasks.

Note: In WebSphere Application Server for z/OS, you must use the Customization Dialog and the jobs it generates to create new cells and nodes. Once you have created a stand-alone application server or Network Deployment cell, however, you use the WebSphere Application Server for z/OS administrative console or scripting to administer it.

The Customization Dialog consists of a set of ISPF panels, file-tailoring skeletons, message libraries, CLISTs and REXX execs that are installed as part of the WebSphere Application Server for z/OS product. It is intended for use under TSO by a systems programmer or WebSphere Application Server for z/OS administrator who is familiar with the z/OS target system on which the resulting WebSphere Application Server for z/OS cells and nodes will run.

The Customization Dialog uses ISPF variables to hold the various values used to create WebSphere Application Server for z/OS customization jobs, scripts and files. See “Customization Dialog variables” on page 5 for more information.

Follow these steps to perform most Customization Dialog customization and migration tasks:

Note: See the *Migrating, coexisting, and interoperating* PDF for information on the migration portion of the Customization Dialog.

1. Start the Customization Dialog. See “Starting the Customization Dialog” for instructions.
2. Choose a set of configuration data sets to hold the generated jobs and files for this task.
3. Set the dialog variables to appropriate values. See “Customization Dialog variables” on page 5 for instructions.
4. Generate the customized jobs, scripts and files based on the dialog variable values you provided, and place them in the configuration data sets.
5. Move the configuration data sets to the z/OS system on which you will perform WebSphere Application Server for z/OS customization or migration tasks.
6. Follow the generated instructions in the .CNTL configuration data set to complete the customization or migration task.

Starting the Customization Dialog

Before you can start the Customization Dialog, you must install the WebSphere® Application Server for z/OS® product and make the product libraries available to the z/OS system on which the Customization Dialog will run. All product libraries should be cataloged. The Customization Dialog does not use the product directory (/usr/lpp/zWebSphere/V6R1 by default), but you must mount it and make it available before the Customization Dialog’s generated jobs can run.

You will need the ability to log on to TSO from a real or emulated 3270-type terminal. Your logon display must support 3270 emulation and be set to a minimum of 32 rows by 80 columns (32 x 80) in order for the ISPF Customization Dialog to run.

- If your terminal has exactly 32 display rows, be sure the PF keys are not shown. These can overlay Customization Dialog input fields. To hide the PF keys, issue the `PFSHOW OFF` command in ISPF.
- If you have a 32-row display and use the ISPF split screen function, deselect “Always show split line” on the ISPF Settings panel and split the screen at the extreme top or bottom of the display. This prevents the split screen line from displaying and lines in the Customization Dialog from being obscured. Other uses of split screen will obscure lines in the Customization Dialog.

The following steps outline how to change your display size setting if you use the IBM® Personal Communications Workstation program. Complete all the steps before you start your TSO session.

1. In the menu bar of the session window, select **Communication**.
2. From the Communication window, select **Configure**.
3. In the window that appears, press the **Session Parameters** button.
4. Use the pull-down menu to select a screen-size setting between 32x80 and 62x160.
5. Press the **OK** button until you are back at the session window.

You will need a TSO user ID that has READ access to the WebSphere Application Server for z/OS product libraries.

- After you have logged on with your user ID, you will see the ISPF menu options. Select **Settings**, and under terminal characteristics find **screen format**, and select **max**. This will maximize your screen for the customization dialogs.

Starting the dialog: The BBOWSTRT command

To start the Customization Dialog, log on to TSO and invoke the BBOWSTRT exec from the SBBOCLIB product library using the following EXEC command.

Note: If you enter the command from TSO Option 6, just type in the command. If you enter the command on the ISPF command line, you must prefix it with ″TSO″.

```
EXEC 'was_hlq.SBBOCLIB(BBOWSTRT)'  'options'
```

where the parameter:

was_hlq
is the data set name high-level qualifier for your WebSphere Application Server for z/OS product libraries.

Note: The dialog libraries (SBBOCLIB, SBBOPxxx, SBBOMxxx, SBBOSLIB and SBBOSLB2) must have the same data set name high-level qualifier in order for the Customization Dialog to allocate the libraries correctly.

Be sure to use the *was_hlq* value that corresponds to the level of WebSphere Application Server for z/OS product code that you will use to run the resulting WebSphere Application Server for z/OS cells.

You can also specify the following options, separated by spaces and surrounded in a single set of quotes:

APPL(*value*)
Specifies the application name you intend to use. This provides for separate sets of Customization Dialog variables saved from one Customization Dialog session to the next. The default value is BBO6. See “Customization Dialog variables” on page 5 for details.

LANG(*value*)
Specifies the national language you want the Customization Dialog to use. Specify ″ENUS″ for English or ″JAPN″ for Japanese. The default value is ENUS.

Note: Some messages will still appear in English even if you specify another national language, using the `LANG` option.

PROD(*list*)

Specifies a list of WebSphere Application Server for z/OS add-on products that are also customized using the dialog. The list should consist of one or more three-character product identifiers from the following table, separated by spaces. If you do not want to customize any add-on products, omit the PROD option.

BBZ	WebSphere Business Integration

PRODHLQ(*list*)

Specifies a list of high-level qualifiers for the WebSphere Application Server for z/OS add-on products you specified with the PROD option. The list should consist of one high-level qualifier for each entry in the PROD list, given in the same order. If you do not want to customize any add-on products, omit the PRODHLQ option.

If you are starting the dialog for the first time or using a new APPL value, you will first see a copyright screen. Press ENTER to continue.

You should then see the Customization Dialog main menu, similar to the following:

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>                                                    Appl: BBO6

   Use this dialog to create WebSphere Application Server for z/OS
   cells and nodes. Specify an option and press Enter.


   1  Configure common MVS Groups and Users

   2  Create stand-alone application server nodes. You must complete
      Option 1 before starting this option.

   3  Create Network Deployment cells and nodes. You must complete
      Option 1 before starting this option.

   4  Migrate a node.

   5  License and Notices.
```

If you do not see the main menu, review the steps and requirements above.

Customization Dialog variables

The Customization Dialog provides a means to save dialog variable values from one session to the next using the Customization Dialog save files. ISPF can save these variables from one session to the next; in addition, the Customization Dialog can save WebSphere Application Server customization variables in data sets that can be reloaded during later customization sessions.

The WebSphere Application Server for z/OS Customization Dialog uses ISPF variables to store customization values. Although it is possible to reenter these values every time you use the Customization Dialog, this process can be time-consuming and error-prone.

Customization Dialog save files

The Customization Dialog provides its own method for saving and reusing dialog variable values. From most task menus, you can type S on the command line to save the current dialog variables in a file, and type L on the command line to restore dialog variables from a previously-created save file. These files are ordinary z/OS sequential files (RECFM=VB, LRECL=255) with one variable-value pair per line.

Because values in save files are explicitly saved and restored, you are somewhat less likely to use them accidently when customizing a different cell than values stored in the ISPF profile. Also, you can use the save file name to identify the WebSphere Application Server for z/OS cell or node for which they are created, the WebSphere Application Server for z/OS version and release, and so on. You can use a Customization Dialog save file, together with the WebSphere Application Server for z/OS product files, to recreate the WebSphere Application Server for z/OS customization data sets (.CNTL and .DATA) at any time.

One save file that is particularly important is the security domain save file. You should save the security domain values during the creation of each WebSphere Application Server for z/OS security domain and import them whenever you create a new stand-alone application server or Network Deployment cell in that security domain. Within the dialog panels for cell creation, you can view the security domain settings but you can not change them. The panels have specific numbered options for saving the newly created security domain values and restoring them during cell creation so that you do not forget.

Once you have finishing setting customization variable values for a new stand-alone application server or Network Deployment cell, save the variable values before proceeding with customization. This allows you to more easily restart the customization process if you discover you have made an error during customization. You can also import the appropriate saved values when performing tasks, such as federating a stand-alone application server or creating a new managed node, on an existing Network Deployment cell.

The following are some cautions that you should take when working with Customization Dialog save files:

- Customization Dialog save files might contain unencrypted passwords and other sensitive information. Make sure that access to the save files is restricted.
- Save files created with the `S` command include the security domain variables as well. If you need to restore values from both a regular save file and a security domain save file, therefore, restore the security domain save file last.
- Do not edit Customization Dialog save files directly—-instead, restore them in a Customization Dialog session, make the appropriate changes, and save the values again.

You can also use Customization Dialog save files to check for typing errors and similar problems by sorting the save file on the columns containing the dialog variable values. This can help detect typographic errors in product data set names and so on as well as detect ports or UNIX® UID/GID values that are accidently reused within a particular configuration.

The instructions for each configuration task provide guidance on restoring and saving Customization Dialog variable values.

ISPF variable pools

Variable pools are a feature of ISPF that allow the dialog variables for one application to reside separate from those for other applications. Whenever you start the dialog using the ISPSTART command, you can use the `NEWAPPL` option to assign a one to four character name to the application and its variable pool. Variable pool names must be one to four characters long, alphanumeric, and begin with an alphabetic letter. The default variable pool name for the WebSphere Application Server for z/OS Customization Dialog is BBO6.

When you exit ISPF normally, the current editor settings and dialog variable values for each variable pool are saved in a pair of members in the user ID's ISPPROF (ISPF profile) data set. The data set name of the ISPPROF data set is usually something like userid.ISPF.ISPPROF, but it can vary from installation to installation. When saving settings for application *xxxx*, the editor settings are saved in member *xxxx*EDIT and the variable settings are saved in member *xxxx*PROF.

You can use the `APPL` option when starting the WebSphere Application Server for z/OS Customization Dialog to specify the application name (variable pool) that you want to use:

```
EXEC 'was_hlq.SBBOCLIB(BBOWSTRT)' 'APPL(xxxx)'
```

Whenever you start the Customization Dialog for the first time with a particular application name, the product copyright page is displayed before the main menu appears. Each Customization Dialog task menu panel shows the application name in the upper right-hand corner, labelled ″Appl:″.

By using different application names, you can separate variable pools for different WebSphere Application Server for z/OS releases, target systems, or cells. Just be sure to specify the same APPL value whenever you start the Customization Dialog in order to work with a particular WebSphere Application Server for z/OS customization task. You should also use customization save files (described in the following section) to provide long-term, more easily documented backup of customization variable values.

To delete all information from an ISPF variable pool, delete the *xxxx*EDIT and *xxxx*PROF members from the ISPPROF data set.

Note: Ensure when you do this that the ISPF application that uses the variable pool is not active.

If an ISPF dialog such as the WebSphere Application Server for z/OS Customization Dialog terminates abnormally, the variable pool members in the ISPPROF data set will not be updated.

Customization Dialog commands and function keys

The WebSphere Application Server for z/OS Customization Dialog uses only a small number of application-specific primary commands that are listed on the panels on which you can enter them. Of these, the most important are the `S` (Save) and `L` (RESTORE) commands for working with WebSphere Application Server for z/OS customization variable save files. See “Customization Dialog variables” on page 5 for more information.

The following default function keys are used throughout the Customization Dialog:

PF1
Displays a help screen for the current panel.

Note: Within the help panels, slightly different PF key settings are used. Press PF1 from within any help panel for details.

PF2
Sets split screen mode. See “Starting the Customization Dialog” on page 3 for restrictions on split screen mode in the Customization Dialog.

PF3
Exits the current panel.

PF4
Exits the Customization Dialog.

PF7
Scrolls up.

PF8
Scrolls down.

PF9
Swaps screens in split screen mode.

PF10
Scrolls left.

PF11
Scrolls right.

PF12
Retrieves the previous command.

You might also find the following ISPF primary commands useful:

`EPDF`
Allows you to browse or edit a z/OS data set from the ISPF command line

`MSGID`
Turns the display of message identifiers on and off.

`PANELID`
Turns the display of panel names on and off.

`PFSHOW OFF`
Turns off the display of PF key settings.

Note: This is necessary when using a 32-row display.

`PFSHOW ON`
Turns on the display of PF key settings.

`TSO`
Executes a TSO command, CLIST, or REXX command procedure from the command line.

`ZKEYS`
Allows you to display or change the current function key settings.

Creating common MVS groups and users

Perform this task to set up the operating system security prerequisites for a WebSphere Application Server for z/OS cell. This ensures that all servers in the cell are using the same operating system security definitions.

Install the WebSphere Application Server for z/OS product code and review the instructions for using the Customization Dialog. Have available a copy of the worksheet that you completed as part of planning for common groups and users.

You need to perform this task before configuring any application serving environment that uses the common groups and users. If a new WebSphere Application Server for z/OS cell or server on a z/OS system will use the exact same common group and user definitions as an existing server or cell on the same z/OS system, you do not need to repeat this task.

You need to run the jobs generated as part of this task once per security database. If z/OS systems do not share a RACF or other security database, you are responsible for making sure identical security definitions are in place for all WebSphere Application Server for z/OS user IDs, groups, and profiles. See the *Securing applications and their environment* PDF for more information about preparing the security server (RACF).

1. Log on to TSO on the z/OS system on which you intend to configure the common groups and users. Use a user ID that has READ access to the WebSphere Application Server for z/OS product data sets.
2. Start the Customization Dialog. See “Starting the Customization Dialog” on page 3 for details.
3. Choose the configuration data sets in which you will store your customization jobs and data. See “Choosing configuration data sets” on page 9 for details.
4. Set the customization variables according to the values recorded on your common groups and users worksheet. See “Setting the customization variables: Common groups and users” on page 14 for details.
5. Save the common groups and users customization variables in a data set that you will use in later customization steps. See “Saving the common group and user variables” on page 14 for details.
6. Create the customization jobs and files, based on the customization variable values you entered. See “Creating the customization jobs and files” on page 15 for details.

7. Follow the generated customization instructions. See “Following the generated customization instructions: Security domain” on page 17 for details, and a sample set of customization instructions.

You have finished when you have successfully completed the steps in the generated instructions. The common groups and users is in place on the chosen z/OS system. If any z/OS systems that interoperate with or host your planned application serving environment do not share the security database you updated as part of this task, update the security databases of the other systems accordingly.

Proceed with the configuration of the application serving environments that use this common group and user.

Choosing configuration data sets

This article leads you through the ″Allocate target data sets″ option in the Customization Dialog.

You must start the Customization Dialog and select the ″Configure a security domain″ option.

Each option in the Customization Dialog saves customization jobs and files in a pair of customization data sets. While is it possible to reuse these data sets, it is safest to create separate data sets for each WebSphere Application Server for z/OS configuration. The best practice is to use the customization data set name prefix (sometimes referred to as ″*config_hlq*″) to indicate the version and release of WebSphere Application Server for z/OS, the task you are performing, and the cell (and, in some cases, the node name) you are configuring. For example, you might use the following data set name prefix for configuring a WebSphere Application Server for z/OS 6.0.1 security domain for cell PRODCELL:

SYSPROG1.WAS601.PRODCELL.SECD

Complete this task before generating the customization jobs and files.

1. On the main dialog panel, type the appropriate number in the *Option* field to select ″Allocate target data sets″.
2. Press Enter. **Result:** You see a panel that looks similar to the following:

```
-----------------     WebSphere Application Server for z/OS Customization      ------------------
Option  ===>

 Allocate Target Data Sets

 Specify a high level qualifier (HLQ) and press Enter to allocate the
 data sets to contain the generated jobs and instructions. You can
 specify multiple qualifiers (up to 39 characters).

 High level qualifier:                                           .CNTL
                                                                 .DATA

 The Dialog will display data set allocation panels. You can make
 changes to the default allocations, however you should not change
 the DCB characteristics of the data sets.

    .CNTL  - a PDS with fixed block 80-byte records to
             contain customization jobs.

    .DATA  - a PDS with variable length data to contain
             other data produced by the Customization Dialog.
```

3. Fill in your chosen configuration data set name prefix value (*config_hlq*). If the data sets *config_hlq*.CNTL and *config_hlq*.DATA do not exist, you will be prompted for data set allocation information. If the data sets already exist, a message will inform you that they will be reused.

The data sets *config_hlq*.CNTL and *config_hlq*.DATA are allocated and will store customization jobs and files. These data set names will also be saved along with the customization variables.

Security domains and the customization dialog

A security domain provides cell-level granularity of security permissions, which:

- Provides cell-level granularity of roles
- Allows different administrators to be assigned for test and production
- Is also used as the APPL profile for servers in the cell

Configure a security domain by using the customization dialog to customize your settings. This provides:

- A set of saved variables to be used when creating base and deployment manager configurations
- A new sample set of Resource Access Control Facility (RACF) customization jobs that must only be run once when the domain is created

Security information is stored in a new SavedVariables file because security domain information can span multiple cells (including test and production). You should be able to use the existing variables you have defined previously. (Make sure you save the values and record the location where it is saved.)

The RACF profiles that are created and checked differently because of this are:

- CBIND
- EJBROLE
- APPL

Tip: When setting up base application server, save the general customization variables and security domain variables to different files.

Use CBIND profiles to restrict access to servers if no other specific profile is set. If there is no security domain identifier, enter the following RACF commands:

```
/*  CBIND profiles in case no server definition is set        */
"RDEFINE CBIND CB.BIND.* UACC(NONE)"
"RDEFINE CBIND CB.* UACC(NONE)"
```

If there is a security domain identifier defined as TESTSYS, enter:

```
/*  CBIND CB.BIND.domain_name.                                 */
"RDEFINE CBIND CB.BIND.TESTSYS.* UACC(NONE)"
"RDEFINE CBIND CB.TESTSYS.* UACC(NONE)"
```

Use an APPL profile to protect WebSphere Application Server for z/OS. Sample profiles can grant APPL access to everyone if you use the universal access authority, UACC(NONE), and grant access to the configuration group, unauthenticated user IDs, and all valid WebSphere Application Server for z/OS user IDs.

For example, if there is no security domain, enter the following RACF commands:

```
RDEFINE APPL CB390 UACC(NONE)
PERMIT CB390 CLASS(APPL) ID(TSCLGP) ACCESS(READ)
```

And if there is a security domain identifier defined as TESTSYS, for example, enter:

```
RDEFINE APPL TESTSYS UACC(NONE)
PERMIT TESTSYS CLASS(APPL) ID(TSCLGP) ACCESS(READ)
```

EJBROLE profiles are defined for role-based authorization checks if there is no security domain identifier and the configuration group is defined as TSTCFG. Note that these are default values set at bootstrap, which is the minimum set of users requiring access to naming and administrative roles for a Local OS registry when System Authorization Facility (SAF) authorization is selected.

The following roles must be defined for both operating system and application security. Enter the following RACF commands:

```
RDEFINE EJBROLE administrator UACC(NONE)
RDEFINE EJBROLE monitor       UACC(NONE)
RDEFINE EJBROLE configurator UACC(NONE)
RDEFINE EJBROLE  operator     UACC(NONE)
RDEFINE EJBROLE  deployer     UACC(NONE)
RDEFINE EJBROLE  AdminSecurityManager     UACC(NONE)

PERMIT administrator  CLASS(EJBROLE)  ID(TSTCFG) ACCESS(READ)
PERMIT monitor        CLASS(EJBROLE)  ID(TSTCFG) ACCESS(READ)
PERMIT configurator   CLASS(EJBROLE)  ID(TSTCFG) ACCESS(READ)
PERMIT operator       CLASS(EJBROLE)  ID(TSTCFG) ACCESS(READ)
PERMIT deployer       CLASS(EJBROLE)  ID(TSTCFG) ACCESS(READ)
PERMIT AdminSecurityManager       CLASS(EJBROLE)  ID(TSTCFG) ACCESS(READ)

/* Setting up EJBRoles Profiles for Naming roles             */
RDEFINE EJBROLE CosNamingRead   UACC(NONE)
PERMIT CosNamingRead  CLASS(EJBROLE)  ID(TSGUEST) ACCESS(READ)
RDEFINE EJBROLE CosNamingWrite  UACC(NONE)
RDEFINE EJBROLE CosNamingCreate UACC(NONE)
RDEFINE EJBROLE CosNamingDelete UACC(NONE)
```

If there is a security domain identifier defined as TESTSYS and the configuration group is defined as TSTCFG, enter the following RACF commands:

```
RDEFINE EJBROLE TESTSYS.administrator UACC(NONE)
RDEFINE EJBROLE TESTSYS.monitor       UACC(NONE)
RDEFINE EJBROLE TESTSYS.configurator  UACC(NONE)
RDEFINE EJBROLE TESTSYS.operator      UACC(NONE)
RDEFINE EJBROLE TESTSYS.deployer      UACC(NONE)
RDEFINE EJBROLE TESTSYS.AdminSecurityManager     UACC(NONE)

PERMIT TESTSYS.administrator  CLASS(EJBROLE)  ID(TSTCFG) ACCESS(READ)
PERMIT TESTSYS.monitor        CLASS(EJBROLE)  ID(TSTCFG) ACCESS(READ)
PERMIT TESTSYS.configurator   CLASS(EJBROLE)  ID(TSTCFG) ACCESS(READ)
PERMIT TESTSYS.operator       CLASS(EJBROLE)  ID(TSTCFG) ACCESS(READ)
PERMIT TESTSYS.deployer       CLASS(EJBROLE)  ID(TSTCFG) ACCESS(READ)
PERMIT TESTSYS.AdminSecurityManager        CLASS(EJBROLE)  ID(TSTCFG) ACCESS(READ)

/* Setting up EJBRoles Profiles for Naming roles         */
RDEFINE EJBROLE TESTSYS.CosNamingRead   UACC(NONE)
PERMIT TESTSYS.CosNamingRead  CLASS(EJBROLE)  ID(TSGUEST) ACCESS(READ)
RDEFINE EJBROLE TESTSYS.CosNamingWrite  UACC(NONE)
RDEFINE EJBROLE TESTSYS.CosNamingCreate UACC(NONE)
RDEFINE EJBROLE TESTSYS.CosNamingDelete UACC(NONE)
```

Security server definition

A security domain definition in z/OS provides WebSphere Application Server for z/OS with a set of cell-wide z/OS Security Server (RACF) security definitions.

Assign distinct MVS user IDs to servers in isolatable security domains

The user IDs assigned to the control and servant tasks are defined by the SAF STARTED profile. A specific STARTED profile can be created for all controller procedures with a given procedure name. For example, enter the following RACF commands:

```
RDEFINE STARTED TST5ACR.* STDATA(USER(TSSYMCR1) GROUP(TSTCFG) TRACE( YES))
```

Tip: Your controller procedure should be given a unique procedure name for each security domain.

Servant regions use different procedure names. In order to avoid having to customize the security for each new server created, use a generic profile with a generic server prefix (BBO*) to assign the user ID of all servers whose procedure names begin with BBO to a specific profile for other server names by entering the following RACF commands:

```
RDEFINE STARTED BBO*.* STDATA(USER(TSSYMSR1) GROUP(TSTCFG) TRACE(YES))
RDEFINE STARTED TSTS001S.* STDATA(USER(TSSYMSR1) GROUP(TSTCFG) TRACE(YES))
```

Note: You cannot modify the server and generic server prefix name BBO using the customization dialog. If you require isolation and do not want to require security customization when you add new servers, note that there is an implicit relationship between the server name and the MVS user ID of the servant task. If you do not need to use security customization when adding new servers, and you do not require a unique user ID for an authorization request to a particular server, create an alternate STARTED profile with a unique generic server prefix name by entering the following RACF commands:

```
RDEFINE STARTED TST*.* STDATA(USER(TSSYMSR1) GROUP(TSTCFG) TRACE(YES))
```

Each servant procedure in the TEST security domain must be defined with TST as the server prefix name used for all servers in this domain. Create a specific STARTED profile for each server for more control over the authorization of requests on different server by entering the following RACF commands:

```
RDEFINE STARTED TSTS001S.* STDATA(USER(TSSYMSR1) GROUP(TSTCFG) TRACE(YES))
RDEFINE STARTED TSTS002S.* STDATA(USER(TSSYMSR2) GROUP(TSTCFG) TRACE(YES))
```

Restrict SERVER access to security domains. In addition, the server class profiles are used to indicate which servant identities can access the appropriate Workload Manager (WLM) queues WebSphere Application Server for z/OS uses. In order to clearly isolate the security domains sets, note the relationship between server names and servant region MVS user IDs.

The SERVER profile checked is either in the form CB.servername.clustername.cellname or CB.servername.clustername, depending upon whether or not WLM Dynamic Application Environment support is enabled. For example, if your server name is TSTC001, the definitions are set by entering the following RACF commands:

```
RDEFINE SERVER CB.* UACC(NONE)
RDEFINE SERVER CB.*.BBO* UACC(NONE)
RDEFINE SERVER CB.*.BBO*.* UACC(NONE)
RDEFINE SERVER CB.*.TSTC001 UACC(NONE)(READ)
RDEFINE SERVER CB..*.TSTC001.*  UACC(NONE)
```

Permissions to access this server name are given by entering the following RACF commands:

```
PERMIT CB.*.TSTC001 CLASS(SERVER) ID(TSSYMSR1) ACC(READ)
PERMIT CB.*.TSTC001.* CLASS(SERVER) ID(TSSYMSR1) ACC(READ)
```

You can create additional SERVER definitions to accommodate a new server transition prefix, or create specific profiles per servant (similarly to how you can use the STARTED profile). Use a server transition prefix of TST to restrict the access of the TSSYMSR user ID to queues from TST servers. For example, to set this up enter the following RACF commands:

```
RDEFINE SERVER CB.*.TST* UACC(NONE)(READ)
RDEFINE SERVER CB.*.TST.*  UACC(NONE)
PERMIT CB.*.TST* CLASS(SERVER) ID(TSSYMSR1) ACC(READ)
PERMIT CB.*.TST*.* CLASS(SERVER) ID(TSSYMSR1) ACC(READ)
```

CBIND profile definitions for servers

If there is no security domain identifier, information is defined during bootstrap. Enter the following RACF commands:

```
RDEFINE CBIND CB.BIND.BBO* UACC(NONE)
RDEFINE CBIND CB.BIND.TSTC001 UACC(NONE)
PERMIT CB.BIND.BBO* CLASS(CBIND) ID(TSTCFG) ACCESS(CONTROL)
PERMIT CB.BIND.TSTC001 CLASS(CBIND) ID(TSTCFG) ACCESS(CONTROL)
RDEFINE CBIND CB.BBO* UACC(NONE)
RDEFINE CBIND CB.TSTC001 UACC(NONE)
```

If there is a security domain identifier defined as TESTSYS, enter:

```
RDEFINE CBIND CB.BIND.TESTSYS.BBO* UACC(NONE)
RDEFINE CBIND CB.BIND.TESTSYS.TSTC001 UACC(NONE)
PERMIT CB.BIND.TESTSYS.BBO* CLASS(CBIND) ID(TSTCFG) ACCESS(CONTROL)
PERMIT CB.BIND.TESTSYS.TSTC001 CLASS(CBIND) ID(TSTCFG) ACCESS(CONTROL)
RDEFINE CBIND CB.TESTSYS.BBO* UACC(NONE)
RDEFINE CBIND CB.TESTSYS.TSTC001 UACC(NONE)
```

Note:

- If you wish to create a new specific server with a prefix other than BBO*, define a specific CBIND profile by entering the following RACF commands:

  ```
  RDEFINE CBIND CB.BIND.TSTC002 UACC(NONE)
  PERMIT CB.BIND.TSTC002 CLASS(CBIND) ID(TSTCFG) ACCESS(CONTROL)
  RDEFINE CBIND CB.TSTC002 UACC(NONE)
  ```

- The samples create server definitions with specific server names (but a generic profile with a server prefix of BBO). If you have created an alternative server prefix and wish to avoid additional CBIND definitions, add generic CBIND profiles that reflect the new name by entering the following RACF commands:

  ```
  RDEFINE CBIND CB.BIND.TESTSYS.TST* UACC(NONE)
  PERMIT CB.BIND.TESTSYS.TST* CLASS(CBIND) ID(TSTCFG) ACCESS(CONTROL)
  RDEFINE CBIND CB.TESTSYS.TST* UACC(NONE)
  ```

Refer to Planning a security domain for more information on security domains.

Note:

- Different security domains should have different users and groups. While the domain identifier separates the RACF classes (CBIND, EJBROLE, APPL), it does not separate the file permissions for configuration files in the Hierarchical File System (HFS). For example, if:
  - The administrator is WSADMIN in group WSCFG
  - The Servant region identity is WASSRV (which must also belong to the WSCFG group)
  - The user TOM has READ access to the TEST.administrator EJBROLE but not to the PROD.administrator EJBROLE,

  TOM cannot use the administration application to make changes to the PROD cell.
- A rogue application running in the TEST application server can modify HFS files in the PROD cell. This is because the TEST server runs with the WASSRV user ID that belongs to the WSCFG group. Both the TEST and PROD HFS files can be modified by the WSCFG group. For maximum protection, PROD should be created and associated with a different RACF group from TEST.

Security customization dialog settings:

The Customization Dialog enables you to create a security domain for your WebSphere Application Server for z/OS configuration.

Note:

- You must set up a base Application Server using the dialogs before using this one to set up a Network Deployment node, which is managed by the deployment manager process (dmgr). It is critical that you **LOAD** saved environment variables from the base Application Server into the deployment manager node that federates the base node. Do this before performing security customization on the deployment manager node.
- If the APPL class is active and you have defined a profile for WebSphere Application Server, make sure that all z/OS identities using WebSphere Application Server services have READ permission to the WebSphere Application Server APPL profile. This includes all WebSphere Application Server identities, WebSphere Application Server unauthenticated identities, WebSphere Application Server administrative identities, user IDs based on role-to-user mappings, and all user identities for system users. If you have not defined a security domain, the

APPL profile used is CBS390 or the name used as the security domain identifier. If you have defined a security domain, the APPL profile used is the security domain name.

- When adding an administrator to the administrative console using local operating system security, if the APPL class is activated, the administrator's user ID must be authorized to the CBS390 (or the name specified as the security domain identifier) APPL class for RACF as well. If the administrator's user ID is not authorized to CBS390 APPL, message BBOS0108E is issued, indicating that the credential-handling function (RunAsGetSpecCred) failed in routine because the user is not authorized.

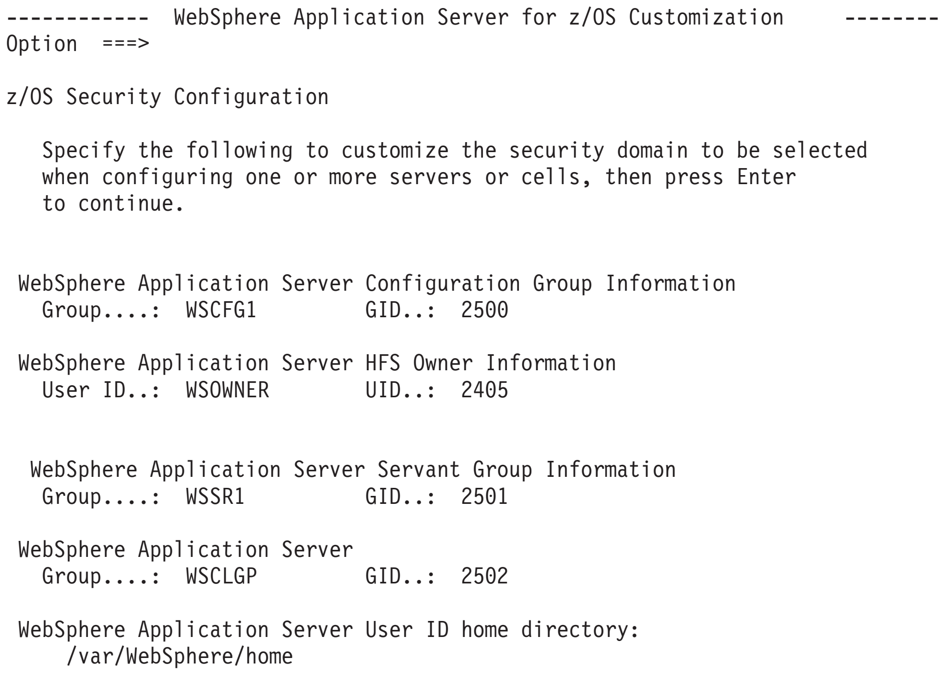
Setting the customization variables: Common groups and users

This article describes how to complete the ″Define variables″ option for a WebSphere Application Server for z/OS security domain.

You must start the Customization Dialog and select the ″Create stand-alone Application Server nodes″ option. Have the Common MVS Groups and Users Customization Dialog worksheet completed and at hand. A copy of this worksheet is available in the *Installing your application serving environment* PDF.

1. On the ″Configure common groups and users″ panel, type the appropriate number in the *Option* field to select ″Define variables″ and press **Enter**.
2. Fill in the ″Common groups and users define variables″ panels using the following screen shots and tips as your guides. When you are done with each panel, press **Enter**.

   Common groups and users define variables panel



```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

z/OS Security Configuration

   Specify the following to customize the security domain to be selected
   when configuring one or more servers or cells, then press Enter
   to continue.


 WebSphere Application Server Configuration Group Information
   Group....:  WSCFG1         GID..:  2500

 WebSphere Application Server HFS Owner Information
   User ID..:  WSOWNER        UID..:  2405


  WebSphere Application Server Servant Group Information
   Group....:  WSSR1          GID..:  2501

 WebSphere Application Server
   Group....:  WSCLGP         GID..:  2502

 WebSphere Application Server User ID home directory:
     /var/WebSphere/home
```

Note: The WebSphere Application Server user ID home directory field was added to the Customization Dialog in WebSphere Application Server for z/OS Version 6.0.2.1.

Saving the common group and user variables

You must start the Customization Dialog or Profile Management tool and fill in the common group and user variables.

The common group and user settings are used in the customization of every WebSphere Application Server for z/OS cell. By saving the common group and user variables, you create a saved security configuration that you can use consistently across all nodes in the cells you create for a given security domain.

Complete this task after setting the common group and user variables and before moving on to later customization steps. If you encounter problems during customization and change the common group and user variable values, be sure to re-save them.

Note: This procedure applies to only the ″Configure a common group and user″ dialog option. For information about saving variables for all the other dialog options, see “Saving the cell variables” on page 52

1. On the ″Configure Security Domain″ panel, type the appropriate number in the *Option* field to select ″Save security domain variables″ and press **Enter**. You will see a panel that looks similar to the following:

   ```
   ------------  WebSphere Application Server for z/OS Customization     --------
   Option  ===>

   Save common group and user variables

     Specify the name of a sequential data set to contain the common group and
     user variables, then press Enter to continue. If the data set does not
     exist, the dialog displays the Allocate New Data Set panel, with which you
     can allocate a data set.


    Data set name:
   ```

2. Fill in the name of the sequential data set you will use to hold the common group and user variable values.

   Choose a data set name that identifies the sysplex, cell, or group of cells for which this security domain is defined. If the data set does not exist, you will be prompted for data set allocation information.

3. Record the name of the common group and user variable data set on your common group and user worksheet.

The common group and user settings are saved in the data set you selected.

Creating the customization jobs and files

You must select configuration data sets to use and complete the process of defining variables for this task. See “Choosing configuration data sets” on page 9 and for more information.

The Customization Dialog creates customization batch jobs and data files, based on the variable values you specified in the dialog. The batch jobs and data sets will be written to the *config_hlq*.CNTL and *config_hlq*.DATA configuration data sets that you created with the ″Allocate target data sets″ option.

1. Ensure that the configuration data sets are allocated and not in use.

   Note: Editing a member in *config_hlq*.CNTL or *config_hlq*.DATA will cause this task to fail.

2. On the ″Configure Security Domain″ panel, type the appropriate number in the *Option* field to select ″Generate customization jobs″ and press **Enter**. You will have one of two results:

   - **Result A:** If all variables are defined correctly, you see the ″Specify Job Cards″ panel, which looks similar to this:

     ```
     ------------  WebSphere Application Server for z/OS Customization     --------
     Option  ===>

     Generate Customization Jobs

      This portion of the Customization Dialog generates the jobs you must
     ```

```
run after you complete this Dialog process. You must complete the
customization process before you generate the jobs with this step.
If you have not done this, please return to that step.

Jobs and data files will get generated into data sets:
  'hlq.CNTL'
  'hlq.DATA'
If you wish to generate customization jobs using other data sets, then
exit from this panel and select the "Allocate target data sets" option.

All the jobs that will be tailored for you will need a job card.
Please enter a valid job card for your installation below. The
file tailoring process will update the jobname for you in all the
generated jobs, so you need not be concerned with that portion of
the job cards below. If continuations are needed, replace the
comment cards with continuations.

Specify the job cards, then press Enter to continue.

//jobname  JOB (ACCTNO,ROOM),'userid',CLASS=A,REGION=0M
//*
//*
//*
```

Note: Pay particular attention to the displayed target data sets. Make sure that they are the ones you intend to use.

- **Result B:** If the variables are not defined correctly, you will see the ″Verification″ panel. Decide whether the warnings or errors are serious enough to warrant returning to the ″Define variables″ option.

  Note: If the return code is 8 or greater, return to the ″Define variables″ option and fix the uncovered problems. If you saved the variables previously, be sure to re-save them after making any updates.

3. Fill in the job card information, according to your installation requirements. For each job, the dialog generates a jobname and the ″JOB″ keyword to match the member name of the PDS, but you specify the rest.

   Note: If you need to run these jobs on a particular system in the sysplex (for example, JES2 MAS or JES3 complex), you should specify the necessary Scheduling Environment (SCHENV), JES2 JOBPARM, or JES3 //*MAIN statement at this time.

   Example of a job card entry:

   ```
   //jobname JOB 1234,USER1,NOTIFY=????,MSGCLASS=O,REGION=0M
   //*            USER=SYSADM1,PASSWORD=SYSADM1
   /*JOBPARM SYSAFF=SYSB
   ```

   Note: This example is useful for jobs that require a user ID other than that of the logged-on TSO user. (This is typically a user ID with UID=0.) In that case, you can just put a comma at the end of the first line, put in the correct user ID on the second line, then uncomment that second line.

   You might want to use RACF SUBMIT authority to avoid having to keep passwords in your configuration data sets.

4. Fix any errors. If there are errors anywhere, you will see the ″Error″ panel. Press PF3 to exit the error panel, then enter the correct panel to fix the errors. Then return to the ″Generate Customization Jobs″ option and pick up where you left off. If necessary, you can update the variables and rerun this option. The generation process will delete and re-tailor all the members.

   Note: Compress the configuration data sets before you rerun this option.

You are done when all the jobs are generated. You can move ahead to viewing the generated jobs. See “Following the generated customization instructions: Security domain” on page 17 for more information.

Following the generated customization instructions: Security domain

You must generate the customization jobs and files for this task.

The Customization Dialog creates a set of instructions for each customization task. Follow these instructions to tailor and customize a security domain on your system.

Note: Do not attempt to fix a typographical error or make a change by modifying the generated output. Many of the variables are used in multiple members of the target data sets; if you do not change them all, therefore, you will run into problems that are very difficult to diagnose.

1. On the ″Configure Security Domain″ panel, type the appropriate number in the *Option* field to select ″View instructions″ and press **Enter**. ISPF Browse will open and you will see the BBOSDINS member of *config_hlq*.CNTL.
2. Read the instructions carefully, both to preview the customization process and to find any typographical or other errors you might have made while entering the customization variable values.
3. Follow the instructions as given. There are two ways to follow the directions:
   - Follow the instructions while remaining in ISPF Browse.
   - Record the data set name and member at the top of the screen and either print the instructions or use ISPF split screen and browse or edit the instructions while you follow them.
4. Fix any problems. If you encounter problems caused by your Customization Dialog values, modify your variables using the dialog, regenerate the instructions, and restart the customization process.

   Note: Remember that you cannot generate new customization jobs while either configuration data set is open!

You are done with this customization task when you have successfully followed the generated instructions.

Sample generated instructions: Common MVS groups and users:

This article presents a sample of what the Customization Dialog's generated instructions might look like. This is a sample only--you must use the instructions generated from your own variables when configuring your system.

```
-----------------------------------------------
Instructions for customizing a WebSphere for z/OS common MVS groups and
users.

The customization dialog has created jobs based on the information you
provided. These instructions tell you how to modify the operating
system and run the jobs to customize WebSphere for z/OS.

RULES:

1.  If you created the target data sets (*.CNTL and *.DATA) on another
    (driving) system, you must copy them to the target system and give
    them the same data set names.

2.  You must perform these instructions on your target system.

3.  You will have saved the z/OS security definition values
    in a data set. These values will need to be loaded and used
    when creating a stand-alone application server or a Network
    Deployment environment.

    ------------------------------------------------------------------

Running the customized jobs
---------------------------

The customization dialog built a number of batch jobs with the
variables you supplied. You must run the jobs in the order listed
```

```
below using user IDs with the appropriate authority.

The customization dialog for WebSphere for z/OS does not attempt to
update configuration data for your base operating system or existing
subsystems.

BEFORE YOU BEGIN: You must copy the target data sets (*.CNTL and
*.DATA) to your target system and give them the same data set names,
and you must be running on your target system.

Follow the table below, which lists in order the jobs you must submit
and the commands you must enter. Special handling notes are included
in the table. All jobs are members of

HOLLOW.WASBB61.CNTL.

Attention: After submitting each job, carefully check the output.
Errors may exist even when all return codes are zero.

+-----------+---------------------------------------------------------+
|           | The following three jobs (BBOSBRAJ, BBOSBRAK, BBOSBRAM)  |
|           | do not need to be run if the indicated groups, user IDs |
|           | and directories already exist with the correct gid,     |
|           | uid and ownership permission values, as given below.    |
+-----------+---------------------------------------------------------+
| BBOSBRAJ  | User ID requirement: Authority to update data set       |
+-----------+                                                         |
| Done:     | USERID.WASBB61.DATA.                                    |
|           |                                                         |
|           | This job builds (but does not execute) the RACF commands|
| By:       | to create common WebSphere for z/OS groups and user IDs:|
|           |                                                         |
|           |  Configuration group:   WSCFG1 (gid 2500)               |
|           |  Servant group:         WSSR1 (gid 2501)                |
|           |  Local user group:      WSCLGP (gid 2502)               |
|           |  HFS owner user ID:     WSOWNER (uid 2405)              |
|           |                                                         |
|           | The commands are placed into member BBOSBRAK of data set|
|           | USERID.WASBB61.DATA.                                    |
|           |                                                         |
|           | Carefully review these definitions with your security   |
|           | administrator.                                          |
+-----------+---------------------------------------------------------+
| BBOSBRAK  | User ID requirement: RACF special authority.            |
+-----------+                                                         |
| Done:     | This job executes the RACF commands created by the      |
|           | previous job.  If the group and user IDs named above    |
|           | have already been created during a previous WebSphere   |
|           | for z/OS configuration and are in all target system RACF|
|           | database(s), you do not need to rerun this job.         |
|           |                                                         |
|           | RESULT: You may receive errors, such as INVALID USER    |
|           | messages, from this job because a user ID, group  or    |
|           | profile is already defined.  Make sure the existing     |
|           | user ID, group or profile has the same characteristics  |
|           | as the user ID, group or profile being created by       |
|           | BBOSBRAK.  If not, then change the values in the        |
|           | customization dialog which are causing the conflict,    |
|           | regenerate the customization jobs, and restart the      |
|           | process.                                                |
|           |                                                         |
|           | When this step is complete, all groups and user IDs     |
|           | listed above for job BBOSBRAJ should be defined in the  |
|           | RACF database on each target system for the cell.       |
|           | Note: the WAS owner user ID WSOWNER MUST have the WAS   |
|           | configuration group WSCFG1 as its default OMVS group.   |
|           |                                                         |
| By:       |                                                          |
|           |                                                          |
+-----------+----------------------------------------------------------+
| BBOSBRAM  | User ID requirement: UID=0.                              |
+-----------+                                                          |
| Done:     | This job creates home directories for WebSphere for z/OS |
|           | user IDS. These home directories will be subdirectories  |
|           | of /var/WebSphere/home                                   |
| By:       |                                                          |
|           | This job will:                                           |
|           |                                                          |
|           | Create the following directory with permission bits 755: |
|           |  /var/WebSphere/home                                     |
|           |                                                          |
|           |  Create the following directory with ownership           |
|           |  WSOWNER:WSCFG1 and permission bits 770:                 |
|           |                                                          |
|           |   /var/WebSphere/home/WSCFG1                             |
|           |                                                          |
|           |  Create the following directory with ownership           |
|           |  WSOWNER:WSSR1 and permission bits 770:                  |
|           |                                                          |
|           |   /var/WebSphere/home/WSSR1                              |
|           |                                                          |
|           |  Create the following directory with ownership           |
|           |  WSOWNER:WSCLGP and permission bits 770:                 |
|           |                                                          |
|           |   /var/WebSphere/home/WSCLGP                             |
|           |                                                          |
|           | This job should be run on each z/OS system that will     |
|           | host WebSphere Application Server nodes using these      |
|           | WebSphere for z/OS common groups and owner user ID.      |
|           | After execution, verify that the directories have been   |
|           | created with the correct permissions on each system.     |
|           |                                                          |
|           | If these directories already exist with the specified   |
|           | ownership and permission on a target system, then this   |
|           | job does not need to be run on that system.              |
|           |                                                          |
|           | ATTENTION: If the directory                              |
|           |  /var/WebSphere/home                                     |
|           | is used by applications other than WebSphere Application |
|           | Server, make sure that the permissions set by            |
|           | BBOSBRAM (755) are appropriate, or change them manually. |
|           | This directory must be world-readable for Websphere      |
|           | Application Server to run correctly.                     |
+-----------+----------------------------------------------------------+

+-----------+----------------------------------------------------------+
```

Creating a stand-alone application server cell

This article leads you through the tasks involved in setting up a WebSphere Application Server for z/OS stand-alone application server environment.

Ensure that the security domain was successfully created on the z/OS target system for the new stand-alone application server. Have available a copy of the worksheet that you completed as a part of your planning for a stand-alone application server cell.

Follow the steps below to set up a new WebSphere Application Server for z/OS stand-alone application server cell.

1. Log on to TSO on the z/OS system on which you intend to configure the stand-alone application server cell. Use a user ID that has READ access to the WebSphere Application Server for z/OS product data

sets. You will also need access to a user ID with authority to make security system updates and a user ID with UID 0. (These can all be the same user ID.)

2. Start the Customization Dialog. See “Starting the Customization Dialog” on page 3 for details.
3. Choose the configuration data sets in which you will store your customization jobs and data. See “Choosing configuration data sets” on page 21 for details.
4. Load the security domain variables saved from the security domain you intend to use for this cell. See “Loading the common MVS groups and users variables” on page 21 for details.
5. Set the customization variables according to the values recorded on your stand-alone application server worksheet. See “Setting the customization variables: Stand-alone application server cell” on page 22 for details.
6. (Optional but recommended.) Save the stand-alone application server customization variables in a data set. See “Saving the cell variables” on page 52 for details.
7. Create the customization jobs and files, based on the customization variable values you entered. See “Creating the customization jobs and files” on page 31 for details.
8. Follow the generated customization instructions. See “Following the generated customization instructions: Stand-alone application server cell” on page 33 for details, and a sample set of customization instructions.

You are done when you have successfully completed the steps in the generated instructions. The new stand-alone application server is up and running on the chosen z/OS system. See “Working with your new server” on page 44 for more information. You can now deploy and test applications on your new stand-alone application server.

Loading the security domain variables

This article describes how to complete the ″Load security domain variables″ option for a WebSphere Application Server for z/OS stand-alone application server cell.

Create the security domain you will use for the new stand-alone application server node and know the name of the saved security domain configuration variable file that you recorded on the security domain worksheet.

The security domain settings are used in the customization of every WebSphere Application Server for z/OS cell. By loading the security domain variables at the start of node or cell creation, you ensure that the security domain configuration you use is consistent and matches the RACF definitions that have already been set as part of security domain configuration.

Complete this task as the first step in configuring a new stand-alone application server node. If you encounter problems during customization and change the security domain variable values, be sure to re-save them.

1. On the ″Create a stand-alone application server node″ panel, type the appropriate number in the *Option* field to select ″Load security domain variables″ and press **Enter**. You will see a panel that looks similar to the following:

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Load Security Domain Variables

 Specify the name of a data set containing the security domain variables,
 then press Enter to continue.

 IBM-supplied defaults are in ''


 Data set name:
```

```
 If this data set is not cataloged, specify the volume.

 Volume:
```

2. Fill in the name of the sequential data set you used to hold the security domain variable values and press **Enter**. The security domain variables will load.

The security domain settings are loaded. You can display these variables, but not change them.

Loading the common MVS groups and users variables

This article describes how to complete the ″Load common MVS groups and users variables″ option for a WebSphere Application Server for z/OS stand-alone application server cell.

Create the common MVS groups and users you will use for the new stand-alone application server node and know the name of the saved security domain configuration variable file that you recorded on the common MVS groups and users worksheet.

The common MVS groups and users settings are used in the customization of every WebSphere Application Server for z/OS cell. By loading the common MVS groups and users variables at the start of node or cell creation, you ensure that the configuration you use is consistent and matches the RACF definitions that have already been set as part of common MVS groups and users configuration.

Complete this task as the first step in configuring a new stand-alone application server node. If you encounter problems during customization and change the common MVS groups and users variable values, be sure to re-save them.

1. On the ″Create a stand-alone application server node″ panel, type the appropriate number in the *Option* field to select ″Load common MVS groups and users variables″ and press **Enter**. You will see a panel that looks similar to the following:

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Load common MVS groups and users variables

 Specify the name of a data set containing the common MVS groups and users
 variables, then press Enter to continue.

 IBM-supplied defaults are in ''


 Data set name:


 If this data set is not cataloged, specify the volume.

 Volume:
```

2. Fill in the name of the sequential data set you used to hold the variable values and press **Enter**. The common MVS groups and users variables will load.

The common MVS groups and users settings are loaded. You can display these variables, but not change them.

Choosing configuration data sets

This article leads you through the ″Allocate target data sets″ option in the Customization Dialog.

You must start the Customization Dialog and select the ″Create stand-alone application server nodes″ option.

Each option in the Customization Dialog saves customization jobs and files in a pair of customization data sets. While is it possible to reuse these data sets, it is safest to create separate data sets for each WebSphere Application Server for z/OS configuration. The best practice is to use the customization data set name prefix (sometimes referred to as ″*config_hlq*″) to indicate the version and release of WebSphere Application Server for z/OS, the task you are performing, and the cell (and, in some cases, the node name) you are configuring.

For example, you might use the following data set name prefix for configuring a WebSphere Application Server for z/OS stand-alone application server with cellname SYSA and servername BB6QA1:

```
SYSPROG.WAS602.SYSA.BB6QA1.SAPPSRVR
```

Complete this task before generating the customization jobs and files.

1. On the main dialog panel, type the appropriate number in the *Option* field to select ″Allocate target data sets″.
2. Press Enter. **Result:** You see a panel that looks similar to the following:

   ```
   -----------------     WebSphere Application Server for z/OS Customization      ------------------
   Option  ===>

    Allocate Target Data Sets

    Specify a high level qualifier (HLQ) and press Enter to allocate the
    data sets to contain the generated jobs and instructions. You can
    specify multiple qualifiers (up to 39 characters).

    High level qualifier:                                         .CNTL
                                                                  .DATA

    The Dialog will display data set allocation panels. You can make
    changes to the default allocations, however you should not change
    the DCB characteristics of the data sets.

       .CNTL  - a PDS with fixed block 80-byte records to
                contain customization jobs.

       .DATA  - a PDS with variable length data to contain
                other data produced by the Customization Dialog.
   ```
3. Fill in your chosen configuration data set name prefix value (*config_hlq*). If the data sets *config_hlq*.CNTL and *config_hlq*.DATA do not exist, you will be prompted for data set allocation information. If the data sets already exist, a message will inform you that they will be reused.

The data sets *config_hlq*.CNTL and *config_hlq*.DATA are allocated and will store customization jobs and files. These data set names will also be saved along with the customization variables.

Setting the customization variables: Stand-alone application server cell

This article describes how to complete the ″Define variables″ option for a WebSphere Application Server for z/OS stand-alone application server node.

You must start the Customization Dialog and select the ″Create stand-alone Application Server nodes″ option. Have the Stand-alone application server cell Customization Dialog worksheet completed and at hand. A copy of this worksheet is available in the *Installing your application serving environment* PDF.

1. On the ″Create a stand-alone application server node″ panel, type the appropriate number in the *Option* field to select ″Define variables″ and press **Enter**.
2. On the ″Define Variables to Configure stand-alone Application Server Node″ panel, type the appropriate number in the *Option* field to select ″System Locations (directories, HLQs, etc.)″ and press **Enter**.
3. Fill in the ″System Locations″ panels using the following screen shots as your guides. When you are done with each panel, press **Enter**.

System Locations

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Locations (1 of 2)

   Specify the following for the system on which you are installing
   WebSphere Application Server for z/OS, then press Enter to continue.
   For some data sets, specify "Y" if they are in STEPLIB.

   System name.:  AQTS      Sysplex name :  MCLXCF01

 Full Names of Data Sets

   PROCLIB:  SYS1.PROCLIB
   PARMLIB:  SYS1.PARMLIB
   SYSEXEC:

   Run WebSphere Application Server from STEPLIB (Y/N)?  Y
   SBBOLPA.:  BOSS.VICOM.W000170.SBBOLPA
   SBBOLOAD:  BOSS.VICOM.W000170.SBBOLOAD
   SBBGLOAD:  BOSS.VICOM.WOOO170.SBBGLOAD
   SBBOLD2.:  BOSS.VICOM.W000170.SBBOLD2
   SBBOEXEC:  BOSS.VICOM.W000170.SBBOEXEC
   SBBOMSG.:  BOSS.VICOM.W000170.SBBOMSG
```

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Locations (2 of 2)

   Specify the following for your customization, then press Enter
   to continue.

 Locations of HFS Resident Components

   WebSphere Application Server product directory:
     /usr/lpp/zWebSphere/V6R1
```

4. On the ″Define Variables to Configure stand-alone Application Server Node″ panel, type the appropriate number in the *Option* field to select ″System Environment Customization″ and press **Enter**.
5. Fill in the ″System Environment Customization″ panels using the following screen shots as your guides. When you have completed each panel, press **Enter**.
   - **System Environment Customization for WebSphere Application Server for z/OS Version 6.0.1**

     ```
     ------------  WebSphere Application Server for z/OS Customization     --------
     Option  ===>

     System Environment Customization (1 of 4)

        Specify the following to customize your system environment, then
        press Enter to continue.

      WebSphere Application Server for z/OS Configuration HFS Information

        Mount point....:  /WebSphere/V6R0
        Name...........:  OMVS.WAS.CONFIG.HFS
        Volume, or '*' for SMS.:  *
        Primary allocation in cylinders...:  250
        Secondary allocation in cylinders.:  100
     ```

     ```
     ------------  WebSphere Application Server for z/OS Customization     --------
     Option  ===>

     System Environment Customization (2 of 4)
     ```

```
   Specify the following to customize your system environment, then
   press Enter to continue.

 WebSphere Error Log Stream Information

        Name.................:  WAS.ERROR.LOG
        Data class ..........:  STANDARD
        Storage class........:
        HLQ for data sets....:  IXGLOGR

   Is log stream CF resident (Y|N):  Y

     If yes, specify structure name.:  WAS_STRUCT
     If no, specify: log stream size:  3000
                     staging size...:  3000

 RRS Log Stream Information

        Group name...........:  MCLXCF01
        Data class...........:  STANDARD
        Storage class........:
        HLQ for data sets....:  IXGLOGR

   Is log stream CF resident (Y|N):  Y

 Create RRS PROC (Y|N).......:  Y
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Environment Customization (3 of 4)

   Specify the following to customize your system environment, then
   press Enter to continue.

 CTRACE Writer Definitions

   Procedure name:  BBOWTR
   User ID.......:  STCRACF
   Group.........:  SYS1

   Trace Data Set Information

     Name...................:  SYS1.AQTS.WAS390.CTRACE
     Volume, or "*" for SMS.:  *
     Primary space in cylinders...:  10
     Secondary space in cylinders.:  0

   Trace Parmlib member suffix...:  60
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Environment Customization (4 of 4)

   Specify the following to customize your server, then press Enter
   to continue.

 Logging Details for Transaction XA Partner Log

 Use Log stream (Y|N):  N

   Log Stream Information

     Name HLQ......................:  WASTXA
     Data class ...................:
     Storage class.................:
```

```
     HLQ for data sets............:  IXGLOGR

     Is log stream CF resident (Y|N):  Y

        If yes, specify structure name.:  WAS_STRUCT
        If no, specify: log stream size:  256
                        staging size...:  256
```

- **System Environment Customization for WebSphere Application Server for z/OS Versions 6.0.2 and Later**

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Environment Customization (1 of 4)

   Specify the following to customize your system environment, then
   press Enter to continue.

 WebSphere Application Server for z/OS Configuration HFS Information

   Mount point....:  /WebSphere/V6R0
   Name...........:  OMVS.WAS.CONFIG.HFS
   Volume, or '*' for SMS.:  *
   Primary allocation in cylinders...:  250
   Secondary allocation in cylinders.:  100
```

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Environment Customization (2 of 4)

   Specify the following to customize your system environment, then
   press Enter to continue.

 WebSphere Error Log Stream Information

        Name.................:  WAS.ERROR.LOG
        Data class ..........:  STANDARD
        Storage class........:
        HLQ for data sets....:  IXGLOGR

   Is log stream CF resident (Y|N):  Y

     If yes, specify structure name.:  WAS_STRUCT
     If no, specify: log stream size:  3000
                     staging size...:  3000
```

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Environment Customization (3 of 4)

   Specify the following to customize your system environment, then
   press Enter to continue.

 CTRACE Writer Definitions

   Trace Parmlib member suffix...:  60
```

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Environment Customization (4 of 4)

   Specify the following to customize your server, then press Enter
   to continue.

 Logging Details for Transaction XA Partner Log
 Use Log stream (Y|N):  N

   Log Stream Information

     Name HLQ......................:  WASTXA
     Data class ...................:
     Storage class.................:
     HLQ for data sets.............:  IXGLOGR

     Is log stream CF resident (Y|N):  Y

        If yes, specify structure name.:  WAS_STRUCT
        If no, specify: log stream size:  256
                        staging size...:  256
```

- **System Environment Customization for WebSphere Application Server for z/OS Versions 6.1**

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Environment Customization (1 of 3)

   Specify the following to customize your system environment, then
   press Enter to continue.

 WebSphere Application Server for z/OS Configuration HFS Information

   Mount point....:  /WebSphere/V6R1
   Name...........:  OMVS.WAS.CONFIG.HFS
   Volume, or '*' for SMS.:  *
   Primary allocation in cylinders...:  250
   Secondary allocation in cylinders.:  100
```

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Environment Customization (2 of 3)

   Specify the following to customize your system environment, then
   press Enter to continue.

 WebSphere Error Log Stream Information

        Log stream Dataset Name:  WAS.ERROR.LOG


 Transaction XA Partner Log Information

   Use Log stream (Y|N):  N

     Log stream HLQ....:    WASTXA
```

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Environment Customization (3 of 3)

   Specify the following to customize your system environment, then
   press Enter to continue.

 CTRACE Writer Definitions

   Trace Parmlib member suffix...:  60
```

6. On the ″Define Variables to Configure stand-alone Application Server Node″ panel, type the appropriate number in the *Option* field to select ″Server Customization″ and press **Enter**.

7. Fill in the ″Server Customization″ panels using the following screen shots as your guides. When you are done with each panel, press **Enter**.

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (1 of 6)

   Specify the following to customize your server, then press Enter
   to continue.

 Application Server Definitions

   WebSphere Application Server home directory:
     /WebSphere/V6R0
         / AppServer

   Cell name (short)......:  AQTS
   Cell name (long).......:  AQTS

   Node name (short)......:  AQTS
   Node name (long).......:  AQTS

   Server name (short)....:  BBOS001
   Server name (long).....:  server1

   Cluster transition name:  BBOC001

   Admin asynch operations procedure name:  BBOW6SH

   Install samples?  (Y/N):  Y
```

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (2 of 6)

   Specify the following to customize your server, then press Enter
   to continue.

 Application Server Definitions

   Controller Information

     Jobname.......:  BBOS001
     Procedure name:  BBO6ACR
     User ID.......:  ASCR1
     UID...........:  2431

   Servant Information

     Jobname.......:  BBOS001S
     Procedure name:  BBO6ASR
     User ID.......:  ASSR1
     UID...........:  2432

   Control Region Adjunct

     Jobname.......:  BBOS001A
     Procedure name:  BBO6CRA
     User ID.......:  ASCRA1
     UID...........:  2433
```

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (3 of 6)

   Specify the following to customize your server, then press Enter
   to continue.

 Application Server Definitions

   Node host name..........:

     SOAP JMX Connector port...............:  8880

   ORB Listener host name..:  *

     ORB port..............................:  2809
     ORB SSL port..........................:  0

   HTTP transport host name:  *

     HTTP port.............................:  9080
     HTTP SSL port.........................:  9443

   Service Integration port...........................:  7276
   Service Integration Secure port....................:  7286
   Service Integration MQ Interoperability port.......:  5558
   Service Integration MQ Interoperability Secure port:  5578
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (4 of 6)

   Specify the following to customize your server, then press Enter
   to continue.

 Application Server Definitions

 Specify your High Availability Manager Host here.  This MUST
 resolve to a single IP address; it cannot be a multihomed host

 High Availability Manager Host:

 High Availability Manager Communication Port:  9353
```

Note: The ″High Availability Manager Host″ field was removed from the Customization Dialog in WebSphere Application Server for z/OS Version 6.0.2.

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (5 of 6)

   Specify the following to customize your server, then press Enter
   to continue.

 Location Service Daemon Definitions

   Daemon home directory:
     /WebSphere/V6R0/Daemon

   Daemon jobname:  BBODMNB

   Procedure name.:  BBO6DMN
   User ID........:  WSDMNCR1
   UID............:  2411

   IP name........:  sdf
   Port...........:  5655
   SSL port.......:  5656

   Register daemon with WLM DNS:  N
```

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (6 of 6)

   (Note:  This panel is optional if you are not configuring a
    database for the Scheduler component)

   Specify the following for the system on which you wish to
   configure your Scheduler database, then press Enter to continue.

 Full Names of Datasets

   SBPXEXEC...........:  SYS1.SBPXEXEC
   DB2 RUNLIB Location:  DB2HLQ.RUNLIB.LOAD

 Scheduler Database Definitions

   DB2 Subsystem Name.:  DSN
   Plan Name..........:  DSNTIA81

   Scheduler Database Name:  SCHEDDB

   Storage Group Name.....:  SYSDEFLT
   Tablespace Name........:  SCHEDTS
   Table Prefix...........:  TBLPRFIX
```

Server Customization

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (1 of 5)

   Specify the following to customize your server, then press Enter
   to continue.

 Application Server Definitions

   WebSphere Application Server home directory:
     /WebSphere/V6R1
         / AppServer

   Cell name (short)......:  AQTS
   Cell name (long).......:  AQTS

   Node name (short)......:  AQTS
   Node name (long).......:  AQTS

   Server name (short)....:  BBOS001
   Server name (long).....:  server1

   Cluster transition name:  BBOC001

   Admin asynch operations procedure name:  BBOW7SH

   WebSphere Application Server Asynchronous Administration Task
      User ID:  WSADMSH        UID:   2504
   Install samples?  (Y/N):  Y
```

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (2 of 5)

   Specify the following to customize your server, then press Enter
   to continue.

 Application Server Definitions
```

```
   Controller Information

     Jobname.......:  BBOS001
     Procedure name:  BBO7ACR
     User ID.......:  ASCR1
     UID...........:  2431

   Servant Information

     Jobname.......:  BBOS001S
     Procedure name:  BBO7ASR
     User ID.......:  ASSR1
     UID...........:  2432

   Control Region Adjunct

     Jobname.......:  BBOS001A
     Procedure name:  BBO7CRA
     User ID.......:  ASCRA1
     UID...........:  2433
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (3 of 5)

   Specify the following to customize your server, then press Enter
   to continue.

 Application Server Definitions

   Node host name..........:  *

     SOAP JMX Connector port...............:  8880

   ORB Listener host name..:  *

     ORB port..............................:  2809
     ORB SSL port..........................:  0

   HTTP transport host name:  *

     Administrative console port...........:  9060
     Administrative console secure port....:  9043
     HTTP transport port...................:  9080
     HTTPS transport port..................:  9443

   High Availability Manger Communication Port........:  9353
   Service Integration port...........................:  7276
   Service Integration Secure port....................:  7286
   Service Integration MQ Interoperability port.......:  5558
   Service Integration MQ Interoperability Secure port:  5578
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (4 of 5)

   Specify the following to customize your server, then press Enter
   to continue.

 Application Server Definitions

 Specify your High Availability Manager Host here.  This MUST
 resolve to a single IP address; it cannot be a multihomed host
 High Availability Manager Host:

 High Availability Manager Communication Port:  9353
```

Note: The ″High Availability Manager Host″ field was removed from the Customization Dialog in WebSphere Application Server for z/OS Version 6.0.2.

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (5 of 5)

   Specify the following to customize your server, then press Enter
   to continue.

 Location Service Daemon Definitions

   Daemon home directory:
     /WebSphere/V6R1/Daemon

   Daemon jobname:  BBODMNB

   Procedure name.:  BBO6DMN
   User ID........:  WSDMNCR1
   UID............:  2411

   IP name........:  xxx
   Port...........:  5655
   SSL port.......:  5656

   Register daemon with WLM DNS:  N
```

8. Fill in the ″Web Server Configuration″ panel using the following screen shot as your guide. When you are done with the panel, press **Enter**.

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

 Web Server Configuration (1 of 1)

 If you are running IBM HTTP Server for z/OS on a local or remote z/OS system
 and wish to have WebSphere Application Server for z/OS manage the
 plugin-cfg.xml file, fill in the following values.

  Web Server Name:  webserver1
   The name used in defining the Web server in the admin console.

  Host.......... :  myhost.acme.com
   IP name or address of the z/OS system on which the Web server is located.

  Port.......... :  80
   HTTP port on which the Web server is listening.

  Application Mapping ((A)ll, (N)one, (D)efault): A
   Determines whether you want to map all applications, none of
   the applications, or the Default Application to the web server
```

9. On the ″Define Variables to Configure stand-alone Application Server Node″ panel, type the appropriate number in the *Option* field to select ″View Security Domain Configuration Panels″ and press **Enter**. These panels display values you previously set in the ″Configure a security domain″ option--you cannot change any of the values here. If you do want to make changes, you must go back to the main dialog panel and run through the ″Configure a security domain″ option again.

Creating the customization jobs and files

You must select configuration data sets to use and complete the process of defining variables for this task. See “Choosing configuration data sets” on page 21 and “Setting the customization variables: Stand-alone application server cell” on page 22 for more information.

The Customization Dialog creates customization batch jobs and data files, based on the variable values you specified in the dialog. The batch jobs and data sets will be written to the *config_hlq*.CNTL and *config_hlq*.DATA configuration data sets that you created with the ″Allocate target data sets″ option.

1. Ensure that the configuration data sets are allocated and not in use.

   Note: Editing a member in *config_hlq*.CNTL or *config_hlq*.DATA will cause this task to fail.

2. On the ″Create a stand-alone application server node″ panel, type the appropriate number in the *Option* field to select ″Generate customization jobs″ and press **Enter**. You will have one of two results:

   - **Result A:** If all variables are defined correctly, you see the ″Specify Job Cards″ panel, which looks similar to this:

     ```
     ------------  WebSphere Application Server for z/OS Customization     --------
     Option  ===>

     Generate Customization Jobs

      This portion of the Customization Dialog generates the jobs you must
      run after you complete this Dialog process. You must complete the
      customization process before you generate the jobs with this step.
      If you have not done this, please return to that step.

      Jobs and data files will get generated into data sets:
        'hlq.CNTL'
        'hlq.DATA'
      If you wish to generate customization jobs using other data sets, then
      exit from this panel and select the "Allocate target data sets" option.

      All the jobs that will be tailored for you will need a job card.
      Please enter a valid job card for your installation below. The
      file tailoring process will update the jobname for you in all the
      generated jobs, so you need not be concerned with that portion of
      the job cards below. If continuations are needed, replace the
      comment cards with continuations.

      Specify the job cards, then press Enter to continue.

      //jobname  JOB (ACCTNO,ROOM),'userid',CLASS=A,REGION=0M
      //*
      //*
      //*
     ```

     Note: Pay particular attention to the displayed target data sets. Make sure that they are the ones you intend to use.

   - **Result B:** If the variables are not defined correctly, you will see the ″Verification″ panel. Decide whether the warnings or errors are serious enough to warrant returning to the ″Define variables″ option.

     Note: If the return code is 8 or greater, return to the ″Define variables″ option and fix the uncovered problems. If you saved the variables previously, be sure to re-save them after making any updates.

3. Fill in the job card information, according to your installation requirements. For each job, the dialog generates a jobname and the ″JOB″ keyword to match the member name of the PDS, but you specify the rest.

   Note: If you need to run these jobs on a particular system in the sysplex (for example, JES2 MAS or JES3 complex), you should specify the necessary Scheduling Environment (SCHENV), JES2 JOBPARM, or JES3 //*MAIN statement at this time.

   Example of a job card entry:

```
//jobname JOB 1234,USER1,NOTIFY=????,MSGCLASS=O,REGION=0M
//*            USER=SYSADM1,PASSWORD=SYSADM1
/*JOBPARM SYSAFF=SYSB
```

Note: This example is useful for jobs that require a user ID other than that of the logged-on TSO user. (This is typically a user ID with UID=0.) In that case, you can just put a comma at the end of the first line, put in the correct user ID on the second line, then uncomment that second line.

You might want to use RACF SUBMIT authority to avoid having to keep passwords in your configuration data sets.

4. Fix any errors. If there are errors anywhere, you will see the ″Error″ panel. Press PF3 to exit the error panel, then enter the correct panel to fix the errors. Then return to the ″Generate Customization Jobs″ option and pick up where you left off. If necessary, you can update the variables and rerun this option. The generation process will delete and re-tailor all the members.

   Note: Compress the configuration data sets before you rerun this option.

You are done when all the jobs are generated. You can move ahead to viewing the generated jobs. See “Following the generated customization instructions: Stand-alone application server cell” for more information.

Following the generated customization instructions: Stand-alone application server cell

You must generate the customization jobs and files for this task.

The Customization Dialog creates a set of instructions for each customization task. Follow these instructions to tailor and customize a stand-alone application server node on your system.

Note: Do not attempt to fix a typographical error or make a change by modifying the generated output. Many of the variables are used in multiple members of the target data sets; if you do not change them all, therefore, you will run into problems that are very difficult to diagnose.

1. On the ″Create a stand-alone application server node″ panel, type the appropriate number in the *Option* field to select ″View instructions″ and press **Enter**. ISPF Browse will open and you will see the BBOSSINS member of *config_hlq*.CNTL.
2. Read the instructions carefully, both to preview the customization process and to find any typographical or other errors that you might have made while entering the customization variable values.
3. Follow the instructions as given.

   There are two ways to follow the directions:
   - Follow the instructions while remaining in ISPF Browse.
   - Record the data set name and member at the top of the screen and either print the instructions or use ISPF split screen and browse or edit the instructions while you follow them.
4. Fix any problems.

   If you encounter problems caused by your Customization Dialog values, modify your variables using the dialog, regenerate the instructions, and restart the customization process.

   Note: Remember that you cannot generate new customization jobs while either configuration data set is open!

You are done with this customization task when you have successfully followed the generated instructions.

Sample generated instructions: Stand-alone application server cell:

This article presents a sample of what the Customization Dialog’s generated instructions might look like. This is a sample only--you must use the instructions generated from your own variables when configuring your system.

Note: This sample is based on the installation of WebSphere Application Server for z/OS Version 6.0.2. Generated instructions from different releases of Version 6 would have different content.

Instructions for customizing WebSphere for z/OS for a stand-alone Application Server node for 6.1.

```
-----------------------------------------------


The customization dialog has created jobs based on the information you
provided. These instructions tell you how to modify the operating
system and run the jobs to customize WebSphere for z/OS.

RULES:

1.  If you created the target data sets (*.CNTL and *.DATA) on another
    (driving) system, you must copy them to the target system and give
    them the same data set names.

2.  You must perform these instructions on your target system.

Doing manual configuration updates
----------------------------------

The customization dialog for WebSphere for z/OS does not attempt to
update configuration data for your base operating system or existing
subsystems. You must do the following manual steps prior to running
the WebSphere for z/OS configuration jobs.

Perform these steps to do manual configuration updates:

1.  Update BLSCUSER. Refer to member BBOIPCSP in

    'MINGLW.TODAY.CNTL'

    In order to use the IPCS support provided by the product, append
    the contents of this member to the BLSCUSER member in your IPCSPARM
    or system PARMLIB datasets.

    ------------------------------------------------------------------

2.  Update SCHEDxx. Refer to member BBOSCHED in

    'USERID.TODAY.CNTL'

    In order to set the correct program properties for the WebSphere
    for z/OS run-time executables, append the contents of this member
    to the SCHEDxx member in your system PARMLIB concatenation.

    Note: When you are finished, issue the command SET SCH=(xx,xx)
    to activate SCHEDxx and load a new program properties table.

    ------------------------------------------------------------------

3.  Make sure the following data sets are APF-authorized:

        DBEL.WASX.SBBOLPA
        DBEL.WASX.SBBOLOAD
        DBEL.WASX.SBBGLOAD
        DBEL.WASX.SBBOLD2

    Add these datasets to your PROGxx or IEAAPFxx parmlib members, as
    appropriate, ensuring you specify the correct volsers.

    ------------------------------------------------------------------

4.  If you want to collect the SMF120 records created by the run-time
    servers, update SMFPRMxx via the following:
```

```
    EXAMPLE:

       SUBSYS(STC,EXITS(IEFU29,IEFACTRT),INTERVAL(SMF,SYNC),
                          TYPE(0,30,70:79,88,89,120,245))
                                                ---

    For details on the SMF records, see related topics in the
    WebSphere for z/OS Information Center at
    http://www.ibm.com/software/webservers/appserv/zos_os390/library/

    ---------------------------------------------------------------------

5.  Update your active BPXPRMxx member to have the following WebSphere
    for z/OS configuration file system:

    OMVS.WAS.CONFIG.HFS

    mounted at:

    /WebSphere/V6R1

    in read/write mode.

    EXAMPLE:

       MOUNT FILESYSTEM('OMVS.WAS.CONFIG.HFS')
         MOUNTPOINT('/WebSphere/V6R1')
          TYPE(HFS)
          MODE(RDWR)

    ---------------------------------------------------------------------

6.  Update TCP/IP by reserving the following ports for WebSphere for
    z/OS:

       SOAP JMX Connector port                             - 8880
       ORB port                                            - 2809
       Administrative console port                         - 9060
       Administrative console secure port                  - 9043

       HTTP Transport port                                 - 9080
       HTTPS Transport port                                - 9443

       High availability manager communication port        - 9353
       Service Integration port                            - 7276
       Service Integration Secure port                     - 7286
       Service Integration MQ Interoperability port        - 5558
       Service Integration MQ Interoperability Secure port - 5578
       Session Initiation Protocol (SIP) port              - 5060
       Session Initiation Protocol (SIP) secure port       - 5061

       Daemon IP port                                      - 5655
       Daemon SSL port                                     - 5656

    View member BBOTCPIP in

    'USERID.TODAY.CNTL'

    Add the contents of this member to the PORT section of the file
    referenced by the DD statement for the TCP/IP profile in the
    TCP/IP start procedure. Cut and paste from this member into the
    data set used by your installation.

    ATTENTION: If another application has already reserved any of these
    ports for its own use, you must resolve the resulting conflict
    before you continue. If you update the WebSphere for z/OS
    customization dialog with new port specifications, be sure to
    regenerate the customization jobs, data, and instructions.

    -------------------------------------------------------------------

7.  The WebSphere product libraries will be placed in STEPLIB as
    needed, rather than in the system link pack area and system link
    list. Make sure that the target MVS system has at least 8MB
    of free storage in extended CSA for the daemon and for EACH
    node (deployment manager node or application server node).

    SBBOLOAD and SBBOLD2:
    ====================
    The following data sets will be placed in the STEPLIB concatenation
    for the location service daemon, controller and servant regions,
    and in the setupCmdLine.sh script in the WebSphere Configuration
    file system. You must not remove these STEPLIB statements.

    DBEL.WASX.SBBOLOAD
    DBEL.WASX.SBBOLD2


    BBORTS61:
    =========

    The BBORTS61 module is used by WebSphere Application Server for
    component trace support. A copy of this module (any maintenance
    level) must be in the system link pack area in order for CTRACE
    to work correctly.

    If a copy of BBORTS61 is currently loaded into LPA, you need take
    no further action.

    Otherwise, issue the following MVS console command to load BBORTS61
    into dynamic LPA:

      SETPROG LPA,ADD,MODNAME=BBORTS61,
              DSNAME=DBEL.WASX.SBBOLPA

    Alternatively, you can place the following statement in a parmlib
    PROGxx member which is activated with the SET PROG= command after
    system IPL is complete:

    LPA ADD MODNAME(BBORTS61)
      DSNAME(DBEL.WASX.SBBOLPA)

    Make sure that the BBORTS61 module is loaded into LPA after each
    system IPL.

    -------------------------------------------------------------------

8.  WebSphere for z/OS customization assumes that the following system
    data sets are in the system link list or link pack area:

    Language Environment     SCEERUN
                             SCEERUN2

    System SSL               SGSKLOAD (z/OS 1.5 and below)
                             SIEALNKE (z/OS 1.6 and above)

    Placing these data sets in the link list or link pack area improves
    performance and insulates your WebSphere for z/OS configuration
    from changes in data set names (for example, when migrating to z/OS
    1.6).

    If the Language Environment or System SSL load module libraries are
    not in your system link list or link pack area, you must perform
    the following steps before starting any WebSphere Application
    Server for z/OS servers:

    - Make sure the data sets are APF-authorized
    - Complete the optional step below to add the data sets to STEPLIB
      in the server JCL and setupCmdLine.sh script(s).

    If you regenerate server cataloged procedures at any point, make
    sure the data sets are added to the new cataloged procedures.

    -------------------------------------------------------------------

9.  If the error logstream
    WAS.ERROR.LOG
    does not already exist on your target system, make a copy of the
    appropriate job in the SBBOJCL data set, customize it according
    to the comments in the job, and run it:

       BBOERRLC    Create an error logstream in a coupling facility
       BBOERRLD    Create a DASD-only error logstream

    -------------------------------------------------------------------

10. WebSphere for z/OS regions open a large number of files (more than
    1024). Make sure your BPXPRMxx parmlib member(s) specify a value of
    MAXFILEPROC that is greater than or equal to 2000. Use the
    following MVS console command to see your current MAXFILEPROC
    setting:

      D OMVS,OPTIONS

    -------------------------------------------------------------------

Running the customized jobs
---------------------------

The customization dialog built a number of batch jobs with the
variables you supplied. You must run the jobs in the order listed
below using user IDs with the appropriate authority.

BEFORE YOU BEGIN: Complete the section above entitled "Doing manual
configuration updates".

Follow the table below, which lists in order the jobs you must submit
and the commands you must enter. Special handling notes are included
in the table. All jobs are members of

USERID.TODAY.CNTL

Attention: After submitting each job, carefully check the output.
Errors may exist even when all return codes are zero.

+-----------+-------------------------------------------------------+
|           | The next three jobs (BBOSBRAJ, BBOSBRAK, BBOSBRAM)    |
|           | do not need to be run if the indicated groups, user IDs |
|           | and directories already exist with the correct gid,   |
|           | uid and ownership permission values, as given below.  |
+-----------+-------------------------------------------------------+
| BBOSBRAJ  | User ID requirement: Authority to update data set     |
+-----------+                                                       |
| Done:     | USERID.TODAY.DATA.                                    |
|           |                                                       |
|           | This job builds (but does not execute) the RACF commands |
| By:       | to create common WebSphere for z/OS groups and user IDs: |
|           |                                                       |
|           |  Configuration group:       WSCFG1 (gid 2500)         |
|           |  Servant group:             WSSR1 (gid 2501)          |
|           |  Local user group:          WSCLGP (gid 2502)         |
|           |  File system owner user ID: WSOWNER (uid 2405)          |
|           |                                                         |
|           | The commands are placed into member BBOSBRAK of data set|
|           | USERID.TODAY.DATA.                                      |
|           |                                                         |
|           | Carefully review these definitions with your security   |
|           | administrator.                                          |
+-----------+---------------------------------------------------------+
| BBOSBRAK  | User ID requirement: RACF special authority.            |
+-----------+                                                         |
| Done:     | This job executes the RACF commands created by the      |
|           | previous job.  If the group and user IDs named above    |
|           | have already been created during a previous WebSphere   |
|           | for z/OS configuration and are in all target system RACF|
|           | database(s), you do not need to rerun this job.         |
|           |                                                         |
|           | RESULT: You may receive errors, such as INVALID USER    |
|           | messages, from this job because a user ID, group  or    |
|           | profile is already defined.  Make sure the existing     |
|           | user ID, group or profile has the same characteristics  |
|           | as the user ID, group or profile being created by       |
|           | BBOSBRAK.  If not, then change the values in the        |
|           | customization dialog which are causing the conflict,    |
|           | regenerate the customization jobs, and restart the      |
|           | process.                                                |
|           |                                                         |
|           | When this step is complete, all groups and user IDs     |
|           | listed above for job BBOSBRAJ should be defined in the  |
|           | RACF database on each target system for the cell.       |
|           | Note: the WAS owner user ID WSOWNER MUST have the WAS   |
|           | configuration group WSCFG1 as its default OMVS group.   |
|           |                                                         |
| By:       |                                                         |
|           |                                                         |
+-----------+---------------------------------------------------------+
| BBOSBRAM  | User ID requirement: UID=0.                             |
+-----------+                                                         |
| Done:     | This job creates home directories for WebSphere for z/OS|
|           | user IDS. These home directories will be subdirectories |
|           | of /var/WebSphere/home                                  |
| By:       |                                                         |
|           | This job will:                                          |
|           |                                                         |
|           | Create the following directory with permission bits 755:|
|           |                                                         |
|           |  /var/WebSphere/home                                    |
|           |                                                         |
|           | Create the following directory with ownership           |
|           | WSOWNER:WSCFG1 and permission bits 770:                 |
|           |                                                         |
|           |  /var/WebSphere/home/WSCFG1                             |
|           |                                                         |
|           | Create the following directory with ownership           |
|           | WSOWNER:WSSR1 and permission bits 770:                  |
|           |                                                         |
|           |  /var/WebSphere/home/WSSR1                              |
|           |                                                         |
|           | Create the following directory with ownership           |
|           | WSOWNER:WSCLGP and permission bits 770:                 |
|           |                                                         |
|           |  /var/WebSphere/home/WSCLGP                             |
|           |                                                         |
|           | This job should be run on each z/OS system that will    |
|           | host WebSphere Application Server nodes using these     |
|           | WebSphere for z/OS common groups and owner user ID.     |
|           | After execution, verify that the directories have been  |
|           | created with the correct permissions on each system.    |
|           |                                                          |
|           | If these directories already exist with the specified    |
|           | ownership and permission on a target system, then this   |
|           | job does not need to be run on that system.              |
|           |                                                          |
|           | ATTENTION: If the directory                              |
|           |  /var/WebSphere/home                                     |
|           | is used by applications other than WebSphere Application |
|           | Server, make sure that the permissions set by            |
|           | BBOSBRAM (755) are appropriate, or change them manually. |
|           | This directory must be world-readable for Websphere      |
|           | Application Server to run correctly.                     |
+-----------+----------------------------------------------------------+
| BBOMSGC   | User ID requirement: Update authority for data set       |
+-----------+ SYS1.MSGENU and/or SYS1.MSGJPN.                          |
| Done:     |                                                          |
|           | ATTENTION: This is optional unless you require message   |
|           | translation.                                             |
| By:       |                                                          |
|           | This job sets up MMS to translate messages for WebSphere |
|           | for z/OS.                                                |
|           |                                                          |
|           | There are two steps to update SYS1.MSGENU and            |
|           | SYS1.MSGJPN. Remove the unneeded step and change the     |
|           | target libraries, if necessary.                          |
+-----------+----------------------------------------------------------+
| BBOCBRAJ  | User ID requirement: Authority to update data set        |
+-----------+                                                          |
| Done:     | USERID.TODAY.DATA                                        |
|           |                                                          |
|           | This job builds (but does not execute) the RACF commands |
| By:       | for the WebSphere for z/OS run-time clusters and places  |
|           | them into member BBOWBRAK of data set                    |
|           |                                                          |
|           | USERID.TODAY.DATA                                        |
|           |                                                          |
|           | Carefully review these definitions with your security    |
|           | administrator.                                           |
+-----------+----------------------------------------------------------+
| BBOCBRAK  | User ID requirement: RACF special authority.             |
+-----------+                                                          |
| Done:     | This job executes the RACF commands set up in the        |
|           | previous job.                                            |
|           |                                                          |
|           | This job creates the WebSphere administrator ID WSADMIN  |
|           | without a password.  You must assign this user ID a      |
|           | password that complies with your institution standards;  |
|           | this is also the password that will be used when logging |
|           | on to the WebSphere Application Server administrative    |
|           | console.                                                 |
|           |                                                          |
|           | Enter the following RACF command to assign a password:   |
|           |                                                          |
|           |   ALTUSER WSADMIN PASSWORD(password) NOEXPIRED           |
|           |                                                          |
|           | If you are using a different security system, make sure  |
|           | that the  user ID has a password.                        |
|           |                                                          |
|           |                                                          |
| By:       | RESULT: You may receive errors, such as INVALID USER     |
|           | messages, from this job because a user ID, group  or     |
|           | profile is already defined.  Make sure the existing      |
|           | user ID, group or profile has the same characteristics   |
|           | as the user ID, group or profile being created by        |
|           | BBOCBRAK.  If not, then change the values in the         |
|           | customization dialog which are causing the conflict,     |
|           | regenerate the customization jobs, and restart the       |
|           | process.                                                |
|           |                                                         |
+-----------+---------------------------------------------------------+
| --------  | Check user ID authorizations.                           |
+-----------+                                                         |
| Done:     | Make sure the WSCFG1 group has read access to all       |
|           | WebSphere product data sets, as well as to any other    |
|           | data sets which will be placed in WebSphere for z/OS    |
|           | cataloged procedure STEPLIB concatenations.             |
|           |                                                         |
|           | Make sure the following user IDs have read access to    |
| By:       | the resolver configuration file in use on your system.  |
|           | Depending on your IP setup, this file may be            |
|           | /etc/resolv.conf, SYS1.TCPPARMS(TCPDATA), or another    |
|           | data set.                                               |
|           |                                                         |
|           | ASCR1                                                   |
|           | ASSR1                                                   |
|           |                                                         |
|           | See the z/OS eNetwork Communication Server IP           |
|           | Configuration manual for the resolver search order.     |
|           |                                                         |
|           | Ensure the following user ID has read access to the data|
|           | sets in your system parmlib concatenation:              |
|           |                                                         |
|           | WSDMNCR1                                                |
|           |                                                         |
|           | ATTENTION:                                              |
|           |                                                         |
|           |  If operator commands are protected by the z/OS security|
|           |  server at your installation, you must ensure that      |
|           |  sufficient authority is given to WebSphere tasks to    |
|           |  control operations.                                    |
|           |                                                         |
|           |  The Application Server Controller user ID (ASCR1)      |
|           |  needs the ability to perform operations on started     |
|           |  tasks belonging to WebSphere Application Server for    |
|           |  z/OS.                                                  |
|           |                                                         |
|           |  The asynchronous administrator user ID, and any user   |
|           |  ID used to run the federation job when the node agent  |
|           |  is started automatically, need the authority to issue  |
|           |  the MVS START command.                                 |
|           |                                                         |
|           |  If you are currently controlling MVS console command   |
|           |  authority with SAF OPERCMDS profiles, grant the        |
|           |  following authorities as indicated, substituting your  |
|           |  own profile names:                                     |
|           |                                                         |
|           |  PERMIT  START_profile_name  CLASS(OPERCMDS)            |
|           |          ID (ASCR1  WSADMSH)  ACCESS(UPDATE)            |
|           |                                                         |
|           |  PERMIT  STOP_profile_name  CLASS(OPERCMDS)             |
|           |          ID (ASCR1 )  ACCESS(UPDATE)                    |
|           |                                                         |
|           |  PERMIT  MODIFY_profile_name  CLASS(OPERCMDS)           |
|           |          ID (ASCR1 )  ACCESS(UPDATE)                    |
|           |                                                         |
|           |  PERMIT  CANCEL_profile_name  CLASS(OPERCMDS)           |
|           |          ID (ASCR1 )  ACCESS(UPDATE)                    |
|           |                                                         |
|           |  PERMIT  FORCE_profile_name  CLASS(OPERCMDS)            |
|           |          ID (ASCR1 )  ACCESS(UPDATE)                    |
|           |                                                         |
|           |                                                         |
|           |  You must also grant the appropriate console command    |
|           |  authority to any user ID that executes the             |
|           |  startServer.sh or stopServer.sh script.                 |
+-----------+----------------------------------------------------------+
| BBOWCHFS  | User ID requirement: UID=0 and authority to allocate     |
+-----------+                                                          |
| Done:     | OMVS.WAS.CONFIG.HFS                                      |
|           |                                                          |
|           | This job:                                                |
|           |                                                          |
| By:       | o   Creates a mount point directory                      |
|           |                                                          |
|           |     /WebSphere/V6R1                                      |
|           |                                                          |
|           | o   Allocates the configuration file system using        |
|           |     the Hierarchical File System (HFS)                   |
|           |                                                          |
|           |     OMVS.WAS.CONFIG.HFS                                  |
|           |                                                          |
|           |     and mounts it at the above mount point.              |
|           |                                                          |
|           | DO NOT RUN THIS JOB IF:                                  |
|           |   1. The configuration file system already exists and is |
|           |      mounted at the desired mountpoint, or if            |
|           |                                                          |
|           |   2. The mount point directory is controlled by          |
|           |      automount.  Either disable the automount rule for   |
|           |      the configuration mount point while running this    |
|           |      job, or perform the following steps manually:       |
|           |                                                          |
|           |      a. Allocate the configuration file system data set. |
|           |      b. Issue the following shell commands, which will   |
|           |         also cause automount to mount the file system    |
|           |                                                          |
|           |      chmod 775 /WebSphere/V6R1                           |
|           |      chown WSOWNER:WSCFG1 /WebSphere/V6R1                |
|           |                                                          |
|           | BEFORE YOU BEGIN: The BBOWCHFS job assumes your root     |
|           | file system is mounted in read/write mode.  If the root  |
|           | file system is not mounted in read/write mode, manually  |
|           | create the directory                                     |
|           |                                                          |
|           | /WebSphere/V6R1                                          |
|           |                                                          |
|           | and any needed higher directories, set file permissions  |
|           | to 775, and set the owning user ID and group to WSOWNER  |
|           | and WSCFG1 before running BBOWCHFS.                      |
|           |                                                          |
|           | EXAMPLE: If you plan to use /WebSphere/V6R0 as your      |
|           | directory, issue the following commands from within the  |
|           | OMVS shell:                                              |
|           |                                                          |
|           |   mkdir -p -m 775 /WebSphere/V6R0                        |
|           |   chown -R WSOWNER:WSCFG1 /WebSphere                     |
|           |                                                          |
+-----------+----------------------------------------------------------+
| BBOWHFSA  | User ID requirement: UID=0.                              |
+-----------+                                                          |
| Done:     | This job populates the previously-created file system.   |
|           |                                                          |
|           | Upon completion, examine the job output. Success is      |
| By:       | indicated with a RC=0 in the job output.                 |
|           |                                                          |
+-----------+----------------------------------------------------------+
| BBOWCPY1  | User ID requirement: update authority for:               |
+-----------+                                                          |
| Done:     |                                                          |
|           |                                                          |
|           |     SYS1.PROCLIB                                         |
| By:       |                                                          |
|           |                                                          |
|           |                                                          |
|           | This job copies the tailored start procedures,           |
|           | parameters, and EXECs to the run-time libraries.         |
|           |                                                          |
|           | ATTENTION: Be aware that you may overlay existing        |
|           | members in the above data sets.                          |
|           |                                                          |
+-----------+----------------------------------------------------------+
| BBOWWPFA  | User ID requirement: UID=0.                              |
+-----------+                                                          |
| Done:     | This job sets up the runtime file system.                |
|           |                                                          |
|           |                                                          |
| By:       | Upon completion, examine the job output. Success is      |
|           | indicated by rc=0.                                       |
|           |                                                          |
|           | Note: If the BBOWWPFA (profile creation) job fails,      |
|           | delete the WAS_HOME/profiles/default directory and all   |
|           | its contents before rerunning BBOWWPFA.                  |
|           |                                                          |
+-----------+----------------------------------------------------------+
| BBOWHFSB  | User ID requirement: UID=0.                              |
+-----------+                                                          |
| Done:     | This job will complete the file system initialization.   |
|           |                                                          |
|           | Upon completion, examine the job output. Success is      |
| By:       | indicated with a RC=0 in the job output.                 |
|           |                                                          |
+-----------+----------------------------------------------------------+
| --------  | All WebSphere Application Server processes require       |
+-----------+ access to the Language Environment and System SSL load   |
| Done:     | modules.                                                 |
|           |                                                          |
|           | If the SCEERUN, SCEERUN2 and System SSL load module      |
|           | libraries are not in the system link list or link pack   |
|           | area, add them to the STEPLIB DD concatenation in each   |
|           | of the following cataloged procedures in                 |
|           | SYS1.PROCLIB:                                            |
|           |                                                          |
|           |     BBO6ACRZ                                             |
|           |     BBO6ASRZ                                             |
|           |     BBO6CRAZ                                             |
|           |     BBO6DMNZ                                             |
|           |                                                          |
|           | and also add the full data set names, separated by       |
|           | colons (:), to the STEPLIB variable in the shell script  |
|           |                                                          |
|           |     /WebSphere/V6R1/                                     |
|           |      AppServer/                                          |
|           |       profiles/default/bin/setupCmdLine.sh               |
| By:       |                                                          |
|           | When modifying the setupCmdLine.sh script, do not        |
|           | remove lines or comment them out, as this may cause      |
|           | problems with automated updates to the script.           |
|           |                                                          |
|           | Add only those data sets which are NOT in the link list  |
|           | or link pack area.                                       |
|           |                                                          |
+-----------+----------------------------------------------------------+
| --------  | Make sure Resource Recovery Services (RRS) is active.    |
+-----------+ (See the InfoCenter for setup instructions if necessary.)|
| Done:     | Look for the following console message to verify that    |
|           | RRS was successfully started:                            |
|           |                                                          |
| By:       |                                                          |
|           |   ASA2011I RRS INITIALIZATION COMPLETE. COMPONENT         |
|           |     ID=SCRRS                                              |
|           |                                                           |
+-----------+-----------------------------------------------------------+
| --------  | If your system is busy, you may want to include a rule    |
+-----------+ in your WLM policy that OMVS work for job BBOS001         |
| Done:     | (such as the postinstaller step) is to run in a service   |
|           | class with a high service objective.                      |
| By:       |                                                           |
+-----------+-----------------------------------------------------------+
| --------  | Start the Application Server                              |
+-----------+                                                           |
| Done:     | Issue the MVS command                                     |
|           |                                                           |
|           |   START BBO6ACR,JOBNAME=BBOS001,                          |
|           |         ENV=AQFT.AQFT.BBOS001                             |
|           |                                                           |
| By:       | This command starts the Application Server. Wait until    |
|           | the server is finished initializing before proceeding.    |
|           |                                                           |
|           | RESULT: The following message appears on the console and  |
|           | in the job log of                                         |
|           |                                                           |
|           | BBOS001                                                   |
|           |                                                           |
|           |   BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR     |
|           |     z/OS CONTROL PROCESS BBOS001                          |
+-----------+-----------------------------------------------------------+
| BBOWIVT   | User ID requirement: UID=0 or WSADMIN                     |
+-----------+                                                           |
| Done:     | This job runs the IVT application. See related topics in  |
|           | the WebSphere for z/OS Information Center at              |
|           | http://www.ibm.com/software/webservers/appserv/zos_os390/ |
|           | library/                                                  |
|           |                                                           |
|           | for information about how to run this job.                |
| By:       |                                                           |
+-----------+-----------------------------------------------------------+
| The product is now configured and verified.                           |
|                                                                       |
| To start the application server, issue the MVS command:               |
|                                                                       |
|   START BBO6ACR,JOBNAME=BBOS001,                                      |
|         ENV=AQFT.AQFT.BBOS001                                         |
|                                                                       |
| To stop the application server, enter the MVS command:                |
|                                                                       |
|   STOP BBODMNB                                                        |
+-----------------------------------------------------------------------+
```

The following is a useful script that helps you define security controls for clusters. It is in data set

'USERID.TODAY.DATA'

```
+-----------+-----------------------------------------------------------+
| BBOWBRAC  | This is a sample exec you can modify to include           |
+-----------+ installation-specific RACF controls. This exec defines    |
| Done:     | all the user IDs and groups that are necessary and        |
|           | sufficient for installing WebSphere for z/OS.             |
|           |                                                           |
| By:       | Additionally, there are commented sections for other      |
|           | components that might be used (SSL, for example).         |
+-----------+-----------------------------------------------------------+
```

Working with your new server

Once you complete the customization instructions, you will have a WebSphere Application Server for z/OS stand-alone application server. The application server includes a (simple) cell and node structure. This article provides useful information for when you work with your new server.

Before starting your server

Make sure that the WebSphere Application Server for z/OS product file system and configuration file system are mounted. If you chose to load the SBBOLPA (and possibly SBBOLOAD) into the system link pack area, make sure that these libraries are loaded into LPA before starting the server.

Starting the stand-alone application server

To start your stand-alone application server, issue the following MVS™ console command:

```
START server_proc,JOBNAME=server_name,ENV=cell_name.node_name.server_name
```

where:

- *server_proc* is the stand-alone application server controller cataloged procedure.
- *server_name* is the server short name.
- *node_name* is the node short name.
- *cell_name* is the cell short name.

If you chose default values and your system is named MVSA, for example, you would enter the following START command:

```
START BBO6ACR,JOBNAME=BBOS001,ENV=MVSA.MVSA.BBOS001
```

The START command brings up the controller. The controller starts the location service daemon, then uses WLM to start the servant. You should see a message like the following when the entire server is up and running:

```
BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR Z/OS CONTROL PROCESS BBOS001
```

Accessing the server administrative console

Once the server is successfully started, access the administrative console port by pointing a Web browser to the following URL:

```
http://hostname:http_port/ibm/console
```

where:

- *hostname* is the HTTP transport host name you specified during customization.

  Note: If you specified "*" for the HTTP host name, this is actually the node host name.
- *http_port* is the HTTP port you specified during customization.

  Note: The default Administrative console port for the stand-alone application server is 9060. This value has changed since Version 6.0.

Until global security is enabled, you will see a sign-on screen that asks you for a user ID but no password.

The user ID can be anything and is used only to provide basic tracking of changes. Be aware that, until you enable global security, anyone with a Web browser and access to the HTTP port can modify your application serving environment.

You can use the administrative console, scripting, or both to manage the application server and deploy and manage J2EE applications. See the *Using the administrative clients* PDF for more information.

Accessing the Samples Gallery

If you chose to install the sample applications, you can access them by pointing a Web browser to the following URL:

```
http://hostname:http_port/WSsamples/
```

where:

- *hostname* is the HTTP transport host name you specified during customization.

  Note: If you specified "*" for the HTTP host name, this is actually the node host name.
- *http_port* is the HTTP port that you specified during customization.

  Note: The default HTTP port for the stand-alone application server is 9080.

Stopping your stand-alone application server

The easiest way to stop the stand-alone application server is to stop the location service daemon. The location service daemon holds pointers to modules in common storage, and stopping it forces the rest of the cell to shut down. To stop the location service daemon, enter the following MVS console command:

```
STOP daemon_jobname
```

where `daemon_jobname` is the location service daemon jobname. The default location service daemon jobname for a stand-alone application server is BBODMNB.

Using the installation verification test

You initially run the installation verification test (IVT), which verifies that WebSphere Application Server is configured correctly for your system, during ISPF customization of each of your systems. If you want to run the IVT at a time other than during initial customization, however, there are two methods from which you can choose.

Note: These options are now available when you are running a stand-alone application server configuration as well as after federating an application server.

Select either method to invoke the IVT:

- "Running the installation verification test with a job"
- "Running the installation verification test from a command line"

Running the installation verification test with a job:
The application server must be running.

Follow these steps to run the installation verification test (IVT) using the BBOWIVT job.

1. Verify that the application server is running.
2. Confirm that the ivtApp application is installed and started.
3. Submit the job BBOWIVT.

The IVT runs a series of verification tests and reports pass or fail status for each in the messages generated by the BBOWIVT job. The output is written to standard output and to the `WAS_HOME/profiles/default/logs/ivtClient.log` file.

Running the installation verification test from a command line:
The application server must be running.

Follow these steps to run the installation verification test (IVT) from a command line.

1. Verify that the application server is running.
2. Confirm that the ivtApp application is installed and started.
3. From a command line, navigate to the `WAS_HOME/bin` directory.
4. Issue the following command:

   ```
   ivt.sh server_name profile_name -p port_number [-host host_name]
   ```

   where

   - *server_name* is the short name of the server.
   - *profile_name* is the name of the profile.
   - `-p` *port_number* is an argument that specifies the port number.
   - `-host` *host_name* is an optional argument that specifies the host name. If you do not specify a host name, the program will use the host-name value that is set in your TCP/IP hosts file.

   Example:

   ```
   /WebSphere/V6R0/AppServer/bin> ivt.sh serverj default -p 9080 -host myhost
   ```

The IVT runs a series of verification tests and reports pass or fail status for each in the messages generated by the BBOWIVT job. The output is written to standard output and to the `WAS_HOME/profiles/default/logs/ivtClient.log` file.

Creating a Network Deployment cell

This article leads you through the tasks involved in setting up a WebSphere Application Server for z/OS Network Deployment environment.

Ensure that the security domain was successfully created on the z/OS target systems for the new Network Deployment cell. Have available a copy of the worksheet that you completed as a part of your planning for a Network Deployment cell.

Perform this task to set up a new WebSphere Application Server for z/OS Network Deployment cell. These steps will set up the cell and the deployment manager node.

1. Log on to TSO on the z/OS system on which you intend to configure the Network Deployment cell's deployment manager. Use a user ID that has READ access to the WebSphere Application Server for z/OS product data sets. You will also need access to a user ID with authority to make security system updates and a user ID with UID 0. (These can all be the same user ID.)
2. Start the Customization Dialog. See "Starting the Customization Dialog" on page 3 for details.
3. Choose the configuration data sets in which you will store your customization jobs and data. See "Choosing configuration data sets" on page 48 for details.
4. Load the security domain variables saved from the security domain you intend to use for this cell. See "Configure common MVS groups and users" on page 47 for details.
5. Set the customization variables according to the values recorded on your Network Deployment cell worksheet. See "Setting the customization variables: Network Deployment cell" on page 49 for details.
6. Save the Network Deployment cell customization variables in a data set. See "Saving the cell variables" on page 52 for details. You will use these variables when creating managed nodes for the cell or federating stand-alone application servers into it.
7. Create the customization jobs and files, based on the customization variable values you entered. See "Creating the customization jobs and files" on page 53 for details.
8. Follow the generated customization instructions. See "Following the generated customization instructions: Network Deployment cell" on page 54 for details, and a sample set of customization instructions.

You are done when you have successfully completed the steps in the generated instructions. The new deployment manager is up and running on the chosen z/OS system. See “Working with your new deployment manager” on page 64 for more information.

Add application server nodes to your cell using one of two methods:

- Create a new managed node using the Customization Dialog and add application servers to it using the administrative console or scripting.
- Federate existing stand-alone application servers into your Network Deployment cell to create managed nodes with application servers.

Loading the security domain variables

This article describes how to complete the ″Load security domain variables″ option for a WebSphere Application Server for z/OS Network Deployment cell.

Create the security domain you will use for the new Network Deployment cell and know the name of the saved security domain configuration variable file that you recorded on the security domain worksheet.

The security domain settings are used in the customization of every WebSphere Application Server for z/OS cell. By loading the security domain variables at the start of node or cell creation, you ensure that the security domain configuration you use is consistent and matches the RACF definitions that have already been set as part of security domain configuration.

Complete this task as the first step in configuring a new Network Deployment cell. If you encounter problems during customization and change the security domain variable values, be sure to re-save them.

1. On the ″Configure Deployment Manager Node″ panel, type the appropriate number in the *Option* field to select ″Load security domain variables″ and press **Enter**. You will see a panel that looks similar to the following:

   ```
   ------------  WebSphere Application Server for z/OS Customization     --------
   Option  ===>

   Load Security Domain Variables

    Specify the name of a data set containing the security domain variables,
    then press Enter to continue.

    IBM-supplied defaults are in ''


    Data set name:


    If this data set is not cataloged, specify the volume.

    Volume:
   ```

2. Fill in the name of the sequential data set you used to hold the security domain variable values and press **Enter**. The security domain variables will load.

The security domain settings are loaded. You can display these variables, but not change them.

Configure common MVS groups and users

This article describes how to complete the ″Configure common MVS groups and users″ option for a WebSphere Application Server for z/OS Network Deployment cell.

Create the common MVS groups and users you will use for the new Network Deployment cell and know the name of the saved configuration variable file that you recorded on the worksheet.

The common MVS groups and users settings are used in the customization of every WebSphere Application Server for z/OS cell. By loading the variables at the start of node or cell creation, you ensure that the configuration you use is consistent and matches the RACF definitions that have already been set as part of the configuration.

Complete this task as the first step in configuring a new Network Deployment cell. If you encounter problems during customization and change the common MVS groups and users variable values, be sure to re-save them.

1. On the ″Configure Deployment Manager Node″ panel, type the appropriate number in the *Option* field to select ″Configure common MVS groups and users″ and press **Enter**. You will see a panel that looks similar to the following:
2. Fill in the name of the sequential data set you used to hold the common MVS group and user variable values and press **Enter**. The variables will load.

The common MVS groups and users settings are loaded. You can display these variables, but not change them.

Choosing configuration data sets

This article leads you through the ″Allocate target data sets″ option in the Customization Dialog.

You must start the Customization Dialog and select the ″Create a Network Deployment cell″ option.

Each option in the Customization Dialog saves customization jobs and files in a pair of customization data sets. While is it possible to reuse these data sets, it is safest to create separate data sets for each WebSphere Application Server for z/OS configuration. The best practice is to use the customization data set name prefix (sometimes referred to as ″*config_hlq*″) to indicate the version and release of WebSphere Application Server for z/OS, the task you are performing, and the cell (and, in some cases, the node name) you are configuring.

For example, you might use the following data set name prefix for configuring a WebSphere Application Server for z/OS Network Deployment cell with cellname AZCELL:

```
SYSPROG.WAS602.AZCELL.NDCONFIG
```

Complete this task before generating the customization jobs and files.

1. On the main dialog panel, type the appropriate number in the *Option* field to select ″Allocate target data sets″.
2. Press Enter. **Result:** You see a panel that looks similar to the following:

```
-----------------     WebSphere Application Server for z/OS Customization      ------------------
Option  ===>

 Allocate Target Data Sets

 Specify a high level qualifier (HLQ) and press Enter to allocate the
 data sets to contain the generated jobs and instructions. You can
 specify multiple qualifiers (up to 39 characters).

 High level qualifier:                                          .CNTL
                                                                .DATA

 The Dialog will display data set allocation panels. You can make
 changes to the default allocations, however you should not change
 the DCB characteristics of the data sets.

    .CNTL  - a PDS with fixed block 80-byte records to
             contain customization jobs.
    .DATA  - a PDS with variable length data to contain
             other data produced by the Customization Dialog.
```

3. Fill in your chosen configuration data set name prefix value (*config_hlq*). If the data sets *config_hlq*.CNTL and *config_hlq*.DATA do not exist, you will be prompted for data set allocation information. If the data sets already exist, a message will inform you that they will be reused.

The data sets *config_hlq*.CNTL and *config_hlq*.DATA are allocated and will store customization jobs and files. These data set names will also be saved along with the customization variables.

Setting the customization variables: Network Deployment cell

This article describes how to complete the ″Define variables″ option for a WebSphere Application Server for z/OS Network Deployment cell.

You must start the Customization Dialog and select the ″Create stand-alone Application Server nodes″ option. Have the Network Deployment cell Customization Dialog worksheet completed and at hand. A copy of this worksheet is available in the *Installing your application serving environment* PDF.

1. On the ″Create a Network Deployment Cell″ panel, type the appropriate number in the *Option* field to select ″Define variables″ and press **Enter**.
2. On the ″Define Variables to Configure Deployment Manager Node″ panel, type the appropriate number in the *Option* field to select ″System Locations (directories, HLQs, etc.)″ and press **Enter**.
3. Fill in the ″System Locations″ panels using the following screen shots as your guides. When you are done with each panel, press **Enter**.

   System Locations

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Locations (1 of 2)

   Specify the following for the system on which you are installing
   WebSphere Application Server for z/OS, then press Enter to continue.
   For some data sets, specify "Y" if they are in STEPLIB.

   System name.:  AQTS      Sysplex name :  MCLXCF01

 Full Names of Data Sets

   PROCLIB:  SYS1.PROCLIB
   PARMLIB:  SYS1.PARMLIB
   SYSEXEC:

   Run WebSphere Application Server from STEPLIB (Y/N)?  Y
   SBBOLPA.:  BOSS.VICOM.W000170.SBBOLPA
   SBBOLOAD:  BOSS.VICOM.W000170.SBBOLOAD
   SBBOLD2.:  BOSS.VICOM.W000170.SBBOLD2
   SBBOEXEC:  BOSS.VICOM.W000170.SBBOEXEC
   SBBOMSG.:  BOSS.VICOM.W000170.SBBOMSG

                                                       Use STEPLIB?
   SCEERUN.:  CEE.SCEERUN                                   N
   SCEERUN2:  CEE.SCEERUN2                                  N
   System SSL: GSK.SGSKLOAD                                 N
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Locations (2 of 2)

   Specify the following for your customization, then press Enter
   to continue.
 Locations of the HFS Resident Components

   WebSphere Application Server product directory:
     /usr/lpp/zWebSphere/V6R1
```

4. On the ″Define Variables to Configure Deployment Manager Node″ panel, type the appropriate number in the *Option* field to select ″System Environment Customization″ and press **Enter**.
5. Fill in the ″System Environment Customization″ panel using the following screen shot as your guide. When you are done, press **Enter**.

System Environment Customization

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Environment Customization (1 of 1)

   Specify the following to customize your system environment, then
   press Enter to continue.

 WebSphere Application Server for z/OS Configuration HFS Information

   Mount point....:  /WebSphere/V6R1
   Name...........:  OMVS.WAS.CONFIG.HFS
   Volume, or '*' for SMS.:  *
   Primary allocation in cylinders...:  250
   Secondary allocation in cylinders.:  100
```

6. On the ″Define Variables to Configure Deployment Manager Node″ panel, type the appropriate number in the *Option* field to select ″Server Customization″ and press **Enter**.
7. Fill in the ″Server Customization″ panels using the following screen shots as your guides. When you are done with each panel, press **Enter**.

Server Customization

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (1 of 4)

   Specify the following to customize your server, then press Enter
   to continue.

 Deployment Manager Definitions

   WebSphere Application Server home directory:
     /WebSphere/V6R1
         / DeploymentManager

   Cell name (short)......:  MCLXCF01
   Cell name (long).......:  MCLXCF01Network

   Node name (short)......:  MCLXCF01
   Node name (long).......:  MCLXCF01Manager

   Server name (short)....:  BBODMGR
   Server name (long).....:  dmgr

   Cluster transition name:  BBODMGR

   Add ability to administer Proxy Server(s) (Y/N) ?:  N
```

Note: The ″Add ability to administer Proxy Server(s) (Y/N)″ field was added to the Customization Dialog in WebSphere Application Server for z/OS Version 6.0.2.

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (2 of 4)
```

```
   Specify the following to customize your server, then press Enter
   to continue.

 Deployment Manager Definitions

   Controller Information

     Jobname.......:  BBODMGR
     Procedure name:  BBO6DCR
     User ID.......:  DMCR1
     UID...........:  2421

   Servant Information

     Jobname.......:  BBODMGRS
     Procedure name:  BBO6DSR
     User ID.......:  DMSR1
     UID...........:  2422
------------  WebSphere Application Server for z/OS Customization     -------
Option  ===>

Server Customization (3 of 4)

   Specify the following to customize your server, the press Enter
   to continue.

 Deployment Manager Definitions

   Node host name..........:

     SOAP JMX connector port...................:  8879
     Cell Discovery Address port...............:  7277
     DRS CLIENT Address port...................:  7989

   ORB Listener host name..:  *

     ORB port..................................:  9809
     ORB SSL port..............................:  0

   HTTP transport host name:  *

     HTTP port.................................:  9060
     HTTP SSL port.............................:  9043

 The High Availability Manager Host MUST resolve to a single
 IP address.  It cannot be a multihomed host.

   High Availability Manager Host:

   High Availability Manager Communication Port:  9352
```

Note: The ″High Availability Manager Host″ field was removed from the Customization Dialog in WebSphere Application Server for z/OS Version 6.0.2.

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (4 of 4)

   Specify the following to customize your server, then press Enter
   to continue.

 Location Service Daemon Definitions

   Daemon home directory:
     /WebSphere/V6R1/Daemon
```

```
  Daemon jobname:  BBODMNC

  Procedure name.:  BBO6DMN
  User ID........:  WSDMNCR1
  UID............:  2411

  IP name........:
  Port...........:  5755
  SSL port.......:  5756

  Register daemon with WLM DNS:  N
```

8. On the ″Define Variables to Configure Deployment Manager Node″ panel, type the appropriate number in the *Option* field to select ″Web Server Configuration″ and press **Enter**.
9. Fill in the ″Web Server Configuration″ panel using the following screen shot as your guide. When you are done with the panel, press **Enter**.

   Web Server Configuration

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

 Security Customization (1 of 1)

  Specify a number and press Enter to customize security for
  WebSphere Application Server for z/OS.  You should review all of the
  variables in the section you select, even if you are using all of the
  IBM-supplied defaults.

  Press PF3 to return to the main menu.

  How do you want to manage user identities and authorization policy?

   The name used in defining the Web server in the admin console.

   1 - Use a z/OS security product.
       The z/OS security product manages users, groups, and
       the authorization policy.

   2 - Use WebSphere Application Server.
       WebSphere Application Server manages users, groups, and
       the authorization policy.

   3 - Do not enable security.
```

10. On the ″Define Variables to Configure Deployment Manager Node″ panel, type the appropriate number in the *Option* field to select ″View Security Customization Configuration Panels″ and press **Enter**.

Saving the cell variables

You must start the Customization Dialog and fill in the variables for the chosen task.

Saving your Network Deployment cell variables allows you to load the same consistent set of values when configuring a new managed node for the cell.

Complete this task after setting the variables for your chosen task. If you encounter problems during customization and change the variable values, be sure to re-save them.

Note: This procedure applies to all dialog options except ″Configure a common group and user.″ For information about saving those variables, see “Saving the common group and user variables” on page 14

1. On the main panel for your chosen task, type ″S″ in the *Option* field to select ″Save customization variables″ and press **Enter**. You will see a panel that looks similar to the following:

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Save Customization Variables

  Specify the name of a sequential data set to contain the
  customization variables, then press Enter to continue. If the
  data set does not exist, the dialog displays the Allocate New
  Data Set panel, with which you can allocate a data set.


 Data set name:
```

2. Fill in the name of the sequential data set you will use to hold the variable values. Choose a data set name that identifies the sysplex, cell or group of cells affected by your chosen task. Enclose the data set name in single quotes. If the data set does not exist, you will be prompted for data set allocation information.
3. Record the name of the data set on the applicable worksheet.

The settings are saved in the data set you selected.

Creating the customization jobs and files

You must select configuration data sets to use and complete the process of defining variables for this task. See “Choosing configuration data sets” on page 48 and “Setting the customization variables: Network Deployment cell” on page 49 for more information.

The Customization Dialog creates customization batch jobs and data files, based on the variable values you specified in the dialog. The batch jobs and data sets will be written to the *config_hlq*.CNTL and *config_hlq*.DATA configuration data sets that you created with the ″Allocate target data sets″ option.

1. Ensure that the configuration data sets are allocated and not in use.

   Note: Editing a member in *config_hlq*.CNTL or *config_hlq*.DATA will cause this task to fail.
2. On the ″Configure Deployment Manager Node″ panel, type the appropriate number in the *Option* field to select ″Generate customization jobs″ and press **Enter**. You will have one of two results:
   - **Result A:** If all variables are defined correctly, you see the ″Specify Job Cards″ panel, which looks similar to this:

     ```
     ------------  WebSphere Application Server for z/OS Customization     --------
     Option  ===>

     Generate Customization Jobs

      This portion of the Customization Dialog generates the jobs you must
      run after you complete this Dialog process. You must complete the
      customization process before you generate the jobs with this step.
      If you have not done this, please return to that step.

      Jobs and data files will get generated into data sets:
        'hlq.CNTL'
        'hlq.DATA'
      If you wish to generate customization jobs using other data sets, then
      exit from this panel and select the "Allocate target data sets" option.

      All the jobs that will be tailored for you will need a job card.
      Please enter a valid job card for your installation below. The
      file tailoring process will update the jobname for you in all the
      generated jobs, so you need not be concerned with that portion of
      the job cards below. If continuations are needed, replace the
      comment cards with continuations.

      Specify the job cards, then press Enter to continue.
     ```

```
//jobname  JOB (ACCTNO,ROOM),'userid',CLASS=A,REGION=0M
//*
//*
//*
```

Note: Pay particular attention to the displayed target data sets. Make sure that they are the ones you intend to use.

- **Result B:** If the variables are not defined correctly, you will see the ″Verification″ panel. Decide whether the warnings or errors are serious enough to warrant returning to the ″Define variables″ option.

  Note: If the return code is 8 or greater, return to the ″Define variables″ option and fix the uncovered problems. If you saved the variables previously, be sure to re-save them after making any updates.

3. Fill in the job card information, according to your installation requirements. For each job, the dialog generates a jobname and the ″JOB″ keyword to match the member name of the PDS, but you specify the rest.

   Note: If you need to run these jobs on a particular system in the sysplex (for example, JES2 MAS or JES3 complex), you should specify the necessary Scheduling Environment (SCHENV), JES2 JOBPARM, or JES3 //*MAIN statement at this time.

   Example of a job card entry:

   ```
   //jobname JOB 1234,USER1,NOTIFY=????,MSGCLASS=O,REGION=0M
   //*            USER=SYSADM1,PASSWORD=SYSADM1
   /*JOBPARM SYSAFF=SYSB
   ```

   Note: This example is useful for jobs that require a user ID other than that of the logged-on TSO user. (This is typically a user ID with UID=0.) In that case, you can just put a comma at the end of the first line, put in the correct user ID on the second line, then uncomment that second line.

   You might want to use RACF SUBMIT authority to avoid having to keep passwords in your configuration data sets.

4. Fix any errors. If there are errors anywhere, you will see the ″Error″ panel. Press PF3 to exit the error panel, then enter the correct panel to fix the errors. Then return to the ″Generate Customization Jobs″ option and pick up where you left off. If necessary, you can update the variables and rerun this option. The generation process will delete and re-tailor all the members.

   Note: Compress the configuration data sets before you rerun this option.

You are done when all the jobs are generated. You can move ahead to viewing the generated jobs. See “Following the generated customization instructions: Network Deployment cell” for more information.

Following the generated customization instructions: Network Deployment cell

You must generate the customization jobs and files for this task.

The Customization Dialog creates a set of instructions for each customization task. Follow these instructions to tailor and customize a Network Deployment cell on your system.

Note: Do not attempt to fix a typographical error or make a change by modifying the generated output. Many of the variables are used in multiple members of the target data sets; if you do not change them all, therefore, you will run into problems that are very difficult to diagnose.

1. On the ″Configure Deployment Manager Node″ panel, type the appropriate number in the *Option* field to select ″View instructions″ and press **Enter**. ISPF Browse will open and you will see the BBOCCINS member of *config_hlq*.CNTL.
2. Read the instructions carefully, both to preview the customization process and to find any typographical or other errors you might have made while entering the customization variable values.

3. Follow the instructions as given. There are two ways to follow the directions:
   - Follow the instructions while remaining in ISPF Browse.
   - Record the data set name and member at the top of the screen and either print the instructions or use ISPF split screen and browse or edit the instructions while you follow them.
4. Fix any problems. If you encounter problems caused by your Customization Dialog values, modify your variables using the dialog, regenerate the instructions, and restart the customization process.

   Note: Remember that you cannot generate new customization jobs while either configuration data set is open!

You are done with this customization task when you have successfully followed the generated instructions.

Sample generated instructions: Network Deployment cell:

This article presents a sample of what the Customization Dialog's generated instructions might look like. This is a sample only--you must use the instructions generated from your own variables when configuring your system.

Note: This sample is based on the installation of WebSphere Application Server for z/OS Version 6.0.2. Generated instructions from different releases of Version 6 would have different content.

Instructions for customizing WebSphere for z/OS for a Deployment Manager node for 6.1.

```
----------------------------------------------

The customization dialog has created jobs based on the information you
provided. These instructions tell you how to modify the operating
system and run the jobs to customize WebSphere for z/OS.

RULES:

1.  If you created the target data sets (*.CNTL and *.DATA) on another
    (driving) system, you must copy them to the target system and give
    them the same data set names.

2.  You must perform these instructions on your target system.

Doing manual configuration updates
----------------------------------

The customization dialog for WebSphere for z/OS does not attempt to
update configuration data for your base operating system or existing
subsystems. You must do the following manual steps prior to running
the WebSphere for z/OS configuration jobs.

Perform these steps to do manual configuration updates:

1.  Update BLSCUSER. Refer to member BBOIPCSP in

    'USERID.TODAY.CNTL'

    In order to use the IPCS support provided by the product, append
    the contents of this member to the BLSCUSER member in your IPCSPARM
    or system PARMLIB datasets.

    ------------------------------------------------------------------

2.  Update SCHEDxx. Refer to member BBOSCHED in

    'USERID.TODAY.CNTL'

    In order to set the correct program properties for the WebSphere
    for z/OS run-time executables, append the contents of this member
    to the SCHEDxx member in your system PARMLIB concatenation.

    Note: When you are finished, issue the command SET SCH=(xx,xx)
    to activate SCHEDxx and load a new program properties table.

    -------------------------------------------------------------------

3.  Make sure the following data sets are APF-authorized:

        DBEL.WASX.SBBOLPA
        DBEL.WASX.SBBOLOAD
        DBEL.WASX.SBBGLOAD
        DBEL.WASX.SBBOLD2

    Add these datasets to your PROGxx or IEAAPFxx parmlib members, as
    appropriate, ensuring you specify the correct volsers.

    -------------------------------------------------------------------

4.  Update your active BPXPRMxx member to have the following WebSphere
    for z/OS configuration file system:

    OMVS.WAS.CONFIG.HFS

    mounted at:

    /WebSphere/V6R1

    in read/write mode.

    EXAMPLE:

       MOUNT FILESYSTEM('OMVS.WAS.CONFIG.HFS')
         MOUNTPOINT('/WebSphere/V6R1')
          TYPE(HFS)
          MODE(RDWR)

    -------------------------------------------------------------------

5.  Update TCP/IP by reserving the following ports for WebSphere for
    z/OS:

       SOAP JMX Connector port                       - 8879
       CELL DISCOVERY ADDRESS port                   - 7277
       ORB port                                      - 9809
       Administrative console port                   - 9060
       Administrative console secure port            - 9043

       High Availability Manager Communications port - 9352

       Daemon IP port                                - 5755
       Daemon SSL port                               - 5756

    View member BBOTCPID in

    'USERID.TODAY.CNTL'

    Add the contents of this member to the PORT section of the file
    referenced by the DD statement for the TCP/IP profile in the
    TCP/IP start procedure. Cut and paste from this member into the
    data set used by your installation.

    ATTENTION: If another application has already reserved any of these
    ports for its own use, you must resolve the resulting conflict
    before you continue. If you update the WebSphere for z/OS
    customization dialog with new port specifications, be sure to
    regenerate the customization jobs, data, and instuctions.
    --------------------------------------------------------------------

6.  The WebSphere product libraries will be placed in STEPLIB as
    needed, rather than in the system link pack area and system link
    list. Make sure that the target MVS system has at least 8MB
    of free storage in extended CSA for the daemon and for EACH
    node (deployment manager node or application server node).

    SBBOLOAD and SBBOLD2:
    ====================
    The following data sets will be placed in the STEPLIB concatenation
    for the location service daemon, controller and servant regions,
    and in the setupCmdLine.sh script in the WebSphere Configuration
    file system. You must not remove these STEPLIB statements.

    DBEL.WASX.SBBOLOAD
    DBEL.WASX.SBBOLD2


    BBORTS61:
    =========

    The BBORTS61 module is used by WebSphere Application Server for
    component trace support. A copy of this module (any maintenance
    level) must be in the system link pack area in order for CTRACE
    to work correctly.

    If a copy of BBORTS61 is currently loaded into LPA, you need take
    no further action.

    Otherwise, issue the following MVS console command to load BBORTS61
    into dynamic LPA:

      SETPROG LPA,ADD,MODNAME=BBORTS61,
              DSNAME=DBEL.WASX.SBBOLPA

    Alternatively, you can place the following statement in a parmlib
    PROGxx member which is activated with the SET PROG= command after
    system IPL is complete:

    LPA ADD MODNAME(BBORTS61)
      DSNAME(DBEL.WASX.SBBOLPA)

    Make sure that the BBORTS61 module is loaded into LPA after each
    system IPL.

    --------------------------------------------------------------------

7.  WebSphere for z/OS customization assumes that the following system
    data sets are in the system link list or link pack area:

    Language Environment     SCEERUN
                             SCEERUN2

    System SSL               SGSKLOAD (z/OS 1.5 and below)
                             SIEALNKE (z/OS 1.6 and above)

    Placing these data sets in the link list or link pack area improves
    performance and insulates your WebSphere for z/OS configuration
    from changes in data set names (for example, when migrating to z/OS
    1.6).

    If the Language Environment or System SSL load module libraries are
    not in your system link list or link pack area, you must perform
    the following steps before starting any WebSphere Application
    Server for z/OS servers:
```

```
    - Make sure the data sets are APF-authorized
    - Complete the optional step below to add the data sets to STEPLIB
      in the server JCL and setupCmdLine.sh script(s).

    If you regenerate server cataloged procedures at any point, make
    sure the data sets are added to the new cataloged procedures.

    ------------------------------------------------------------------

8.  If the error logstream
    WAS.ERROR.LOG
    does not already exist on your target system, make a copy of the
    appropriate job in the SBBOJCL data set, customize it according
    to the comments in the job, and run it:
       BBOERRLC    Create an error logstream in a coupling facility
       BBOERRLD    Create a DASD-only error logstream

    ------------------------------------------------------------------

9.  WebSphere for z/OS regions open a large number of files (more than
    1024). Make sure your BPXPRMxx parmlib member(s) specify a value of
    MAXFILEPROC that is greater than or equal to 2000. Use the
    following MVS console command to see your current MAXFILEPROC
    setting:

      D OMVS,OPTIONS


Running the customized jobs
---------------------------

The customization dialog built a number of batch jobs with the
variables you supplied. You must run the jobs in the order listed
below using user IDs with the appropriate authority.

BEFORE YOU BEGIN: Complete the section above entitled "Doing manual
configuration updates".

Follow the table below, which lists in order the jobs you must submit
and the commands you must enter. Special handling notes are included
in the table. All jobs are members of

USERID.TODAY.CNTL

Attention: After submitting each job, carefully check the output.
Errors may exist even when all return codes are zero.

+-----------+--------------------------------------------------------+
|           | The next three jobs (BBOSBRAJ, BBOSBRAK, BBOSBRAM)     |
|           | do not need to be run if the indicated groups, user IDs|
|           | and directories already exist with the correct gid,    |
|           | uid and ownership permission values, as given below.   |
+-----------+--------------------------------------------------------+
| BBOSBRAJ  | User ID requirement: Authority to update data set      |
+-----------+                                                        |
| Done:     | USERID.TODAY.DATA.                                     |
|           |                                                        |
|           | This job builds (but does not execute) the RACF commands|
| By:       | to create common WebSphere for z/OS groups and user IDs:|
|           |                                                        |
|           |  Configuration group:       WSCFG1 (gid 2500)          |
|           |  Servant group:             WSSR1 (gid 2501)           |
|           |  Local user group:          WSCLGP (gid 2502)          |
|           |  File system owner user ID: WSOWNER (uid 2405)         |
|           |                                                        |
|           | The commands are placed into member BBOSBRAK of data set|
|           | USERID.TODAY.DATA.                                       |
|           |                                                          |
|           | Carefully review these definitions with your security    |
|           | administrator.                                           |
+-----------+----------------------------------------------------------+
| BBOSBRAK  | User ID requirement: RACF special authority.             |
+-----------+                                                          |
| Done:     | This job executes the RACF commands created by the       |
|           | previous job.  If the group and user IDs named above     |
|           | have already been created during a previous WebSphere    |
|           | for z/OS configuration and are in all target system RACF |
|           | database(s), you do not need to rerun this job.          |
|           |                                                          |
|           | RESULT: You may receive errors, such as INVALID USER     |
|           | messages, from this job because a user ID, group  or     |
|           | profile is already defined.  Make sure the existing      |
|           | user ID, group or profile has the same characteristics   |
|           | as the user ID, group or profile being created by        |
|           | BBOSBRAK.  If not, then change the values in the         |
|           | customization dialog which are causing the conflict,     |
|           | regenerate the customization jobs, and restart the       |
|           | process.                                                 |
|           |                                                          |
|           | When this step is complete, all groups and user IDs      |
|           | listed above for job BBOSBRAJ should be defined in the   |
|           | RACF database on each target system for the cell.        |
|           | Note: the WAS owner user ID WSOWNER MUST have the WAS    |
|           | configuration group WSCFG1 as its default OMVS group.    |
|           |                                                          |
| By:       |                                                          |
|           |                                                          |
+-----------+----------------------------------------------------------+
| BBOSBRAM  | User ID requirement: UID=0.                              |
+-----------+                                                          |
| Done:     | This job creates home directories for WebSphere for z/OS |
|           | user IDS. These home directories will be subdirectories  |
|           | of /var/WebSphere/home                                   |
| By:       |                                                          |
|           | This job will:                                           |
|           |                                                          |
|           | Create the following directory with permission bits 755: |
|           |                                                          |
|           |  /var/WebSphere/home                                     |
|           |                                                          |
|           | Create the following directory with ownership            |
|           | WSOWNER:WSCFG1 and permission bits 770:                  |
|           |                                                          |
|           |  /var/WebSphere/home/WSCFG1                              |
|           |                                                          |
|           | Create the following directory with ownership            |
|           | WSOWNER:WSSR1 and permission bits 770:                   |
|           |                                                          |
|           |  /var/WebSphere/home/WSSR1                               |
|           |                                                          |
|           | Create the following directory with ownership            |
|           | WSOWNER:WSCLGP and permission bits 770:                  |
|           |                                                          |
|           |  /var/WebSphere/home/WSCLGP                              |
|           |                                                          |
|           | This job should be run on each z/OS system that will     |
|           | host WebSphere Application Server nodes using these      |
|           | WebSphere for z/OS common groups and owner user ID.      |
|           | After execution, verify that the directories have been   |
|           | created with the correct permissions on each system.     |
|           |                                                          |
|           | If these directories already exist with the specified    |
|           | ownership and permission on a target system, then this   |
|           | job does not need to be run on that system.              |
|           |                                                          |
|           | ATTENTION: If the directory                              |
|           |  /var/WebSphere/home                                     |
|           | is used by applications other than WebSphere Application |
|           | Server, make sure that the permissions set by            |
|           | BBOSBRAM (755) are appropriate, or change them manually. |
|           | This directory must be world-readable for Websphere      |
|           | Application Server to run correctly.                     |
+-----------+----------------------------------------------------------+
| BBODBRAJ  | User ID requirement: Authority to update data set        |
+-----------+                                                          |
| Done:     | USERID.TODAY.DATA                                        |
|           |                                                          |
|           | This job builds (but does not execute) the RACF commands |
|           | for the WebSphere for z/OS run-time clusters and places  |
| By:       | them into member BBODBRAK of data set                    |
|           |                                                          |
|           | USERID.TODAY.DATA                                        |
|           |                                                          |
|           | Carefully review these definitions with your security    |
|           | administrator.                                           |
+-----------+----------------------------------------------------------+
| BBODBRAK  | User ID requirement: RACF special authority.             |
+-----------+                                                          |
| Done:     | This job executes the RACF commands set up in the        |
|           | previous job.                                            |
|           |                                                          |
|           | This job creates the WebSphere administrator ID WSADMIN  |
|           | without a password.  You must assign this user ID a      |
|           | password that complies with your institution standards;  |
|           | this is also the password that will be used when logging |
|           | on to the WebSphere Application Server administrative    |
|           | console.                                                 |
|           |                                                          |
|           | Enter the following RACF command to assign a password:   |
|           |                                                          |
|           |   ALTUSER WSADMIN PASSWORD(password) NOEXPIRED           |
|           |                                                          |
|           | If you are using a different security system, make sure  |
|           | that the WSADMIN user ID has a password.                 |
|           |                                                          |
|           |                                                          |
| By:       | RESULT: You may receive errors, such as INVALID USER     |
|           | messages, from this job because a user ID, group  or     |
|           | profile is already defined.  Make sure the existing      |
|           | user ID, group or profile has the same characteristics   |
|           | as the user ID, group or profile being created by        |
|           | BBODBRAK.  If not, then change the values in the         |
|           | customization dialog which are causing the conflict,     |
|           | regenerate the customization jobs, and restart the       |
|           | process.                                                 |
|           |                                                          |
+-----------+----------------------------------------------------------+
| --------  | Check user ID authorizations.                            |
+-----------+                                                          |
| Done:     | Make sure the WSCFG1 group has read access to all        |
|           | WebSphere product data sets, as well as to any other     |
|           | data sets which will be placed in WebSphere for z/OS     |
|           | cataloged procedure STEPLIB concatenations.              |
|           |                                                          |
|           | Make sure the following user IDs have read access to     |
| By:       | the resolver configuration file in use on your system.   |
|           | Depending on your IP setup, this file may be             |
|           | /etc/resolv.conf, SYS1.TCPPARMS(TCPDATA), or another     |
|           | data set.                                                |
|           |                                                          |
|           | DMCR1                                                    |
|           | DMSR1                                                    |
|           |                                                          |
|           | See the z/OS eNetwork Communication Server IP            |
|           | Configuration manual for the resolver search order.      |
|           |                                                          |
|           | Ensure the following user ID has read access to the data |
|           | sets in your system parmlib concatenation:               |
|           |                                                          |
|           | WSDMNCR1                                                 |
|           |                                                          |
|           | ATTENTION:                                               |
|           |                                                          |
|           |  If operator commands are protected by the z/OS security |
|           |  server at your installation, you must ensure that       |
|           |  sufficient authority is given to WebSphere tasks to     |
|           |  control operations.                                     |
|           |                                                          |
|           |  The Deployment Manager controller user ID (DMCR1)       |
|           |  needs the ability to perform operations on started      |
|           |  tasks belonging to WebSphere Application Server for     |
|           |  z/OS.                                                   |
|           |                                                          |
|           |  The asynchronous administrator user ID, and any user    |
|           |  ID used to run the federation job when the node agent   |
|           |  is started automatically, need the authority to issue   |
|           |  the MVS START command.                                  |
|           |                                                          |
|           |  If you are currently controlling MVS console command    |
|           |  authority with SAF OPERCMDS profiles, grant the         |
|           |  following authorities as indicated, substituting your   |
|           |  own profile names:                                      |
|           |                                                          |
|           |  PERMIT  START_profile_name  CLASS(OPERCMDS)             |
|           |          ID (DMCR1 )  ACCESS(UPDATE)                     |
|           |                                                          |
|           |  PERMIT  STOP_profile_name  CLASS(OPERCMDS)              |
|           |          ID (DMCR1 )  ACCESS(UPDATE)                     |
|           |                                                          |
|           |  PERMIT  MODIFY_profile_name  CLASS(OPERCMDS)            |
|           |          ID (DMCR1 )  ACCESS(UPDATE)                     |
|           |                                                          |
|           |  PERMIT  CANCEL_profile_name  CLASS(OPERCMDS)            |
|           |          ID (DMCR1 )  ACCESS(UPDATE)                     |
|           |                                                          |
|           |  PERMIT  FORCE_profile_name  CLASS(OPERCMDS)             |
|           |          ID (DMCR1 )  ACCESS(UPDATE)                     |
|           |                                                          |
|           |  You must also grant the appropriate console command     |
|           |  authority to any user ID that executes the              |
|           |  startServer.sh or stopServer.sh script.                 |
|           |                                                          |
+-----------+----------------------------------------------------------+
| BBODCHFS  | User ID requirement: UID=0 and authority to allocate     |
+-----------+                                                          |
| Done:     | OMVS.WAS.CONFIG.HFS                                      |
|           |                                                          |
|           | ATTENTION: Skip this step if the mount point is already  |
|           | created, such as for the stand-alone Application Server. |
|           |                                                          |
| By:       | This job:                                                |
|           |                                                          |
|           | o   Creates a mount point directory                      |
|           |                                                          |
|           |     /WebSphere/V6R1                                      |
|           |                                                          |
|           | o   Allocates the configuration file system using        |
|           |     the Hierarchical File System (HFS)                   |
|           |                                                          |
|           |     OMVS.WAS.CONFIG.HFS                                  |
|           |                                                          |
|           |     and mounts it at the above mount point.              |
|           |                                                          |
|           | DO NOT RUN THIS JOB IF:                                  |
|           |   1. The configuration file system already exists and is |
|           |      mounted at the desired mountpoint, or if            |
|           |                                                          |
|           |   2. The mount point directory is controlled by          |
|           |      automount.  Either disable the automount rule for   |
|           |      the configuration mount point while running this    |
|           |      job, or perform the following steps manually:       |
|           |                                                          |
|           |      a. Allocate the configuration file system data set. |
|           |      b. Issue the following shell commands, which will   |
|           |         also cause automount to mount the file system    |
|           |                                                          |
|           |      chmod 775 /WebSphere/V6R1                           |
|           |      chown WSOWNER:WSCFG1 /WebSphere/V6R1                |
|           |                                                          |
|           |                                                          |
|           | BEFORE YOU BEGIN: The BBODCHFS job assumes your root     |
|           | file system is mounted in read/write mode.  If the root  |
|           | file system is not mounted in read/write mode, manually  |
|           | create the directory                                     |
|           |                                                          |
|           | /WebSphere/V6R1                                          |
|           |                                                          |
|           | and any needed higher directories. Set file permissions  |
|           | to 775 and set the owning user ID and group to WSOWNER   |
|           | and WSCFG1 before running BBODCHFS.                      |
|           |                                                          |
|           | EXAMPLE: If you plan to use /WebSphere/V6R0 as your      |
|           | directory, issue the following commands from within the  |
|           | OMVS shell:                                              |
|           |                                                          |
|           |   mkdir -p -m 775 /WebSphere/V6R0                        |
|           |   chown -R WSOWNER:WSCFG1 /WebSphere                     |
|           |                                                          |
+-----------+----------------------------------------------------------+
| BBODHFSA  | User ID requirement: UID=0.                              |
+-----------+                                                          |
| Done:     | This job populates the previously-created file system.   |
|           |                                                          |
|           | Upon completion, examine the job output. Success is      |
| By:       | indicated with a RC=0 in the job output.                 |
|           |                                                          |
+-----------+----------------------------------------------------------+
| BBODCPY1  | User ID requirement: update authority for:               |
+-----------+                                                          |
| Done:     |                                                          |
|           |                                                          |
|           |     SYS1.PROCLIB                                         |
| By:       |                                                          |
|           |                                                          |
|           | This job copies the tailored start procedures,           |
|           | parameters, and EXECs to the run-time product libraries. |
|           |                                                          |
|           | ATTENTION: Be aware that you may overlay existing        |
|           | members in the above data set.                           |
|           |                                                          |
+-----------+----------------------------------------------------------+
| BBOWWPFD  | User ID requirement: UID=0.                              |
+-----------+                                                          |
| Done:     |  This job sets up the runtime file system.               |
|           |                                                          |
|           |                                                          |
| By:       | Upon completion, examine the job output. Success is      |
|           | indicated by rc=0.                                       |
|           |                                                          |
|           | Note: If the BBOWWPFD (profile creation) job fails,      |
|           | delete the WAS_HOME/profiles/default directory and all   |
|           | its contents before rerunning BBOWWPFD.                  |
|           |                                                          |
+-----------+----------------------------------------------------------+
| BBODHFSB  | User ID requirement: UID=0.                              |
+-----------+                                                          |
| Done:     | This job will complete the file system initialization.   |
|           |                                                          |
|           | Upon completion, examine the job output. Success is      |
| By:       | indicated with a RC=0 in the job output.                 |
|           |                                                          |
+-----------+----------------------------------------------------------+
| --------  | All WebSphere Application Server processes require       |
+-----------+ access to the Language Environment and System SSL load   |
| Done:     | modules.                                                 |
|           |                                                          |
|           | If the SCEERUN, SCEERUN2 and System SSL load module      |
|           | libraries are not in the system link list or link pack   |
|           | area, add them to the STEPLIB DD concatenation in each   |
|           | of the following cataloged procedures in                 |
|           | SYS1.PROCLIB:                                            |
|           |                                                          |
|           |     BBO6DCRZ                                             |
|           |     BBO6DSRZ                                             |
|           |     BBO6DMNZ                                             |
|           |                                                          |
|           | and also add the full data set names, separated by       |
|           | colons (:), to the STEPLIB variable in the shell script  |
|           |                                                          |
|           |     /WebSphere/V6R1/                                     |
|           |      DeploymentManager/                                  |
|           |       profiles/default/bin/setupCmdLine.sh               |
| By:       |                                                          |
|           | When modifying the setupCmdLine.sh script, do not        |
|           | remove lines or comment them out, as this may cause      |
|           | problems with automated updates to the script.           |
|           |                                                          |
|           | Add only those data sets which are NOT in the link list  |
|           | or link pack area.                                       |
|           |                                                          |
+-----------+----------------------------------------------------------+
| --------  | Make sure Resource Recovery Services (RRS) is active.    |
+-----------+ (See the InfoCenter for setup instructions if necessary.)|
| Done:     | Look for the following console message to verify that    |
|           | RRS was successfully started:                            |
|           |                                                          |
| By:       |                                                          |
|           |   ASA2011I RRS INITIALIZATION COMPLETE. COMPONENT        |
|           |     ID=SCRRS                                             |
|           |                                                          |
+-----------+----------------------------------------------------------+
| --------  | If your system is busy, you may want to include a rule   |
+-----------+ in your WLM policy that OMVS work for job BBODMGR        |
| Done:     | (such as the postinstaller step) is to run in a service  |
|           | class with a high service objective.                     |
| By:       |                                                          |
+-----------+----------------------------------------------------------+
| --------  | Start the Deployment Manager.                            |
+-----------+                                                          |
| Done:     | Issue the MVS command                                    |
|           |                                                          |
|           |   START BBO6DCR,JOBNAME=BBODMGR,                         |
|           |         ENV=MCLXCF01.MCLXCF01.BBODMGR                    |
|           |                                                          |
| By:       | This command starts the Deployment Manager and also      |
|           | starts the location service daemon. Wait until the       |
|           | server is finished initializing before proceeding.       |
|           |                                                          |
|           | RESULT: The following message appears on the console and |
|           | in the job log of BBODMGR                                |
|           |                                                          |
|           |   BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR    |
|           |     z/OS CONTROL PROCESS BBODMGR                         |
|           |                                                          |
+----------------------------------------------------------------------+
| The product is now configured for a Deployment Manager.              |
|                                                                      |
+-----------+----------------------------------------------------------+

Note: At this point of the configuration, if you want to federate a
managed node or stand-alone Application Server node, you will need to
do the following steps:

Choose option 3  Create Network Deployment cells and nodes, from the
WebSphere Application Server for z/OS Customization main menu

Choose option 3  Federate an existing stand-alone application server
node into an existing Network Deployment cell

+-----------+----------------------------------------------------------+
| To start the Deployment Manager, issue the following MVS command:    |
|                                                                      |
|   START BBO6DCR,JOBNAME=BBODMGR,                                     |
|         ENV=MCLXCF01.MCLXCF01.BBODMGR                                |
|                                                                      |
| To stop the WebSphere for z/OS servers, enter the MVS command:       |
|                                                                      |
|   STOP BBODMNC                                                       |
|                                                                      |
+----------------------------------------------------------------------+

The following is a useful script that helps you define security
controls. It is in data set

'USERID.TODAY.DATA'

+-----------+----------------------------------------------------------+
| BBODBRAC  | This is a sample exec that you may modify to include     |
+-----------+ installation-specific RACF controls. This exec defines   |
| Done:     | all the user IDs and groups that are necessary and       |
|           | sufficient for installing WebSphere for z/OS.            |
|           |                                                          |
| By:       | Additionally, there are commented sections for other     |
|           | components that might be used (SSL, for example).        |
+-----------+----------------------------------------------------------+
```

Working with your new deployment manager

Once you complete the customization instructions, you will have a WebSphere Application Server for z/OS Network Deployment cell. The Network Deployment cell consists of a deployment manager and a location service daemon. (To run J2EE applications, you must add application server nodes. See below for details.) This article provides useful information for working with your new Network Deployment cell.

Before starting your server

Make sure that the WebSphere Application Server for z/OS product HFS and configuration HFS are mounted. If you chose to load the SBBOLPA (and possibly SBBOLOAD) into the system link pack area, make sure that these libraries are loaded into LPA before starting the server.

Starting the deployment manager

To start your deployment manager, issue the following MVS console command:

```
START server_proc,JOBNAME=dmgr_name,ENV=cell_name.node_name.dmgr_name
```

where:

- *server_proc* is the deployment manager controller cataloged procedure.
- *dmgr_name* is the deployment manager short name.
- *node_name* is the deployment manager node short name.
- *cell_name* is the cell short name.

If you chose default values and your system is named MVSA, for example, you would enter the following START command:

```
START BBO6DCR,JOBNAME=BBODMGR,ENV=MVSA.MVSA.BBODMGR
```

The START command brings up the deployment manager controller. The controller starts the location service daemon, then uses WLM to start the deployment manager servant. You should see a message like the following when the deployment manager is up and running:

```
BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR Z/OS CONTROL PROCESS BBODMGR
```

Accessing the server administrative console

Once the deployment manager is successfully started, access the administrative console by pointing a Web browser to the following URL:

```
http://hostname:http_port/ibm/console
```

where:

- *hostname* is the deployment manager HTTP transport host name you specified during customization.

  Note: If you specified "*" for the deployment manager HTTP host name, this is actually the deployment manager node host name.
- *http_port* is the deployment manager HTTP port you specified during customization.

  Note: The default HTTP port for the deployment manager is 9060.

Until global security is enabled, you will see a signon screen that asks you for a user ID but no password.

The user ID can be anything and is used only to provide basic tracking of changes. Be aware that, until you enable global security, anyone with a Web browser and access to the HTTP port can modify your application serving environment.

You can use the administrative console, scripting, or both to manage the Network Deployment cell and deploy and manage J2EE applications. See the *Using the administrative clients* PDF for more information. Before you can deploy applications, however, you need to add application server nodes to your Network Deployment cell.

Adding application server nodes

Application server nodes (also called managed nodes) in a Network Deployment cell consist of a node agent and any number of application servers per node.

Note: Each z/OS system also needs one location service daemon for each stand-alone or Network Deployment cell hosted on the system.

Add an application server node to a Network Deployment cell using one of two methods:

- Create an (empty) managed node using the Customization Dialog. The new node can reside on the same or a different z/OS system as the deployment manager. The new managed node, consisting of just a node agent and perhaps a location service daemon, is federated into the Network Deployment cell. Once this is done, you can use the administrative console or scripting to add application servers and deploy and manage J2EE applications in the node.

  See the section ″Planning for a new managed node in a Network Deployment cell″ in the *Installing your application serving environment* PDF for more information.
- Federate an existing stand-alone application server into the Network Deployment cell. The stand-alone server node becomes a managed node in the Network Deployment cell, along with any J2EE applications that have been deployed on it.

  See the section ″Planning to federate a stand-alone server into a Network Deployment cell″ in the *Installing your application serving environment* PDF for more information.

Stopping your deployment manager

Use one of the following two methods to stop your deployment manager:

- Stop the location service daemon, which also stops the deployment manager and any of the cell’s managed nodes on the same z/OS system. The location service daemon holds pointers to modules in common storage, and stopping it forces the cell’s nodes on the same z/OS system as the location service daemon to shut down. To stop the location service daemon, enter the following MVS console command:

  ```
  STOP daemon_jobname
  ```

  where *daemon_jobname* is the location service daemon jobname. The default location service daemon jobname for a Network Deployment cell is BBODMNC.

  Note: This is the easiest way to stop the deployment manager.
- Stop just the deployment manager, leaving the location service daemon and any managed nodes on the z/OS system still running. This works because the deployment manager is used to administer only the cell--it does not need to be up for J2EE applications in the cell to run. To stop the deployment manager, enter the following MVS console command:

  ```
  STOP dmgr_name
  ```

  where *dmgr_name* is the deployment manager short name. The default deployment manager short name is BBODMGR.

Creating a managed server node

This article leads you through the tasks involved in creating a WebSphere Application Server for z/OS managed server node.

Perform this task to create a new WebSphere Application Server for z/OS managed node.

1. Log on to TSO on the z/OS system on which you intend to configure the managed node. Use a user ID that has READ access to the WebSphere Application Server for z/OS product data sets. You will also need access to a user ID with authority to make security system updates and a user ID with UID 0. (These can all be the same user ID.)
2. Start the Customization Dialog. See “Starting the Customization Dialog” on page 3 for details.

3. Choose the configuration data sets in which you will store your customization jobs and data. See “Choosing configuration data sets” on page 68 for details.
4. Load the security domain variables saved from the security domain you intend to use for this cell. See “Loading the common MVS groups and users variables” on page 68 for details. (Optionally, you can use the L command to load the Network Deployment cell variables if you saved them during Network Deployment cell setup. Network Deployment cell variables include the security domain variables--you do not need to load both.)
5. Set the customization variables according to the values recorded on your managed node worksheet. See “Setting the customization variables: Managed node” on page 69 for details.
6. (Optional but recommended.) Save the managed node customization variables in a data set. See “Saving the cell variables” on page 52 for details.
7. Create the customization jobs and files, based on the customization variable values you entered. See “Creating the customization jobs and files” on page 72 for details.
8. Follow the generated customization instructions. See “Following the generated customization instructions: Managed node” on page 73 for details, and a sample set of customization instructions.

You are done when you have successfully completed the steps in the generated instructions. The new managed node is up and running on the chosen z/OS system. See “Working with your new managed server node” on page 83 for more information.

Loading the security domain variables

This article describes how to complete the ″Load security domain variables″ option for a WebSphere Application Server for z/OS managed node.

Create the security domain you will use for the new managed node and know the name of the saved security domain configuration variable file that you recorded on the security domain worksheet.

The security domain settings are used in the customization of every WebSphere Application Server for z/OS cell. By loading the security domain variables at the start of node or cell creation, you ensure that the security domain configuration you use is consistent and matches the RACF definitions that have already been set as part of security domain configuration.

Complete this task as the first step in configuring a new managed node. If you encounter problems during customization and change the security domain variable values, be sure to re-save them.

1. On the ″Configure Managed Node″ panel, type the appropriate number in the *Option* field to select ″Load security domain variables″ and press **Enter**. You will see a panel that looks similar to the following:

   ```
   ------------  WebSphere Application Server for z/OS Customization     --------
   Option  ===>

   Load Security Domain Variables

    Specify the name of a data set containing the security domain variables,
    then press Enter to continue.

    IBM-supplied defaults are in ''


    Data set name:


    If this data set is not cataloged, specify the volume.

    Volume:
   ```
2. Fill in the name of the sequential data set you used to hold the security domain variable values and press **Enter**. The security domain variables will load.

The security domain settings are loaded. You can display these variables, but not change them.

Loading the common MVS groups and users variables

This article describes how to complete the ″Load common MVS groups and users variables″ option for a WebSphere Application Server for z/OS managed node.

Create the common MVS groups and users you will use for the new managed node and know the name of the saved common MVS groups and users configuration variable file that you recorded on the common MVS group and user worksheet.

The common MVS groups and users settings are used in the customization of every WebSphere Application Server for z/OS cell. By loading the common MVS groups and users variables at the start of node or cell creation, you ensure that the configuration you use is consistent and matches the RACF definitions that have already been set as part of configuration.

Complete this task as the first step in configuring a new managed node. If you encounter problems during customization and change the common MVS groups and users values, be sure to re-save them.

1. On the ″Configure Managed Node″ panel, type the appropriate number in the *Option* field to select ″Common MVS groups and users variables″ and press **Enter**. You will see a panel that looks similar to the following:

   ```
   ------------  WebSphere Application Server for z/OS Customization     --------
   Option  ===>

   Load Common MVS groups and users

    Specify the name of a data set containing the common MVS groups and users
    variables, then press Enter to continue.

    IBM-supplied defaults are in ''


    Data set name:


    If this data set is not cataloged, specify the volume.

    Volume:
   ```

2. Fill in the name of the sequential data set you used to hold the common MVS groups and users variable values and press **Enter**. The variables will load.

The common MVS groups and users settings are loaded. You can display these variables, but not change them.

Choosing configuration data sets

This article leads you through the ″Allocate target data sets″ option in the Customization Dialog.

You must start the Customization Dialog and select the ″Create an empty managed node and add it to an existing Network Deployment cell″ option.

Each option in the Customization Dialog saves customization jobs and files in a pair of customization data sets. While is it possible to reuse these data sets, it is safest to create separate data sets for each WebSphere Application Server for z/OS configuration. The best practice is to use the customization data set name prefix (sometimes referred to as ″*config_hlq*″) to indicate the version and release of WebSphere Application Server for z/OS, the task you are performing, and the cell (and, in some cases, the node name) you are configuring. For example, you might use the following data set name prefix for configuring a WebSphere Application Server for z/OS managed node named MA6N01 for cell MAINT1:

JULIA.WASV6R2.MAINT1.MA6N01.MANAGED

Complete this task before generating the customization jobs and files.

1. On the main dialog panel, type the appropriate number in the *Option* field to select ″Allocate target data sets″.
2. Press Enter. **Result:** You see a panel that looks similar to the following:

   ```
   -----------------     WebSphere Application Server for z/OS Customization      ------------------
   Option  ===>

    Allocate Target Data Sets

    Specify a high level qualifier (HLQ) and press Enter to allocate the
    data sets to contain the generated jobs and instructions. You can
    specify multiple qualifiers (up to 39 characters).

    High level qualifier:                                         .CNTL
                                                                  .DATA

    The Dialog will display data set allocation panels. You can make
    changes to the default allocations, however you should not change
    the DCB characteristics of the data sets.

       .CNTL  - a PDS with fixed block 80-byte records to
                contain customization jobs.

       .DATA  - a PDS with variable length data to contain
                other data produced by the Customization Dialog.
   ```
3. Fill in your chosen configuration data set name prefix value (*config_hlq*). If the data sets *config_hlq*.CNTL and *config_hlq*.DATA do not exist, you will be prompted for data set allocation information. If the data sets already exist, a message will inform you that they will be reused.

The data sets *config_hlq*.CNTL and *config_hlq*.DATA are allocated and will store customization jobs and files. These data set names will also be saved along with the customization variables.

Setting the customization variables: Managed node

This article describes how to complete the ″Define variables″ option for a WebSphere Application Server for z/OS managed node.

You must start the Customization Dialog and select the ″Create stand-alone Application Server nodes″ option. Have the Managed node Customization Dialog worksheet completed and at hand. A copy of this worksheet is available in the *Installing your application serving environment* PDF.

1. On the ″Configure Managed Node″ panel, type the appropriate number in the *Option* field to select ″Define variables″ and press **Enter**.
2. On the ″Define Variables to Configure Managed Node″ panel, type the appropriate number in the *Option* field to select ″System Locations (directories, HLQs, etc.)″ and press **Enter**.
3. Fill in the System Locations panels using the following screen shots as your guides. When you are done with each panel, press **Enter**.

   System Locations

   ```
   ------------  WebSphere Application Server for z/OS Customization     --------
   Option  ===>

   System Locations (1 of 2)

      Specify the following for the system on which you are installing
      WebSphere Application Server for z/OS, then press Enter to continue.
      For some data sets, specify "Y" if they are in STEPLIB.

      System name.:  AQTS      Sysplex name :  MCLXCF01

    Full Names of Data Sets
   ```

```
   PROCLIB:  SYS1.PROCLIB
   PARMLIB:  SYS1.PARMLIB
   SYSEXEC:

   Run WebSphere Application Server from STEPLIB (Y/N)?  Y
   SBBOLPA.:  BOSS.VICOM.W000170.SBBOLPA
   SBBOLOAD:  BOSS.VICOM.W000170.SBBOLOAD
   SBBOLD2.:  BOSS.VICOM.W000170.SBBOLD2
   SBBOEXEC:  BOSS.VICOM.W000170.SBBOEXEC
   SBBOMSG.:  BOSS.VICOM.W000170.SBBOMSG

                                                       Use STEPLIB?
   SCEERUN.:  CEE.SCEERUN                                    N
   SCEERUN2:  CEE.SCEERUN2                                   N
   System SSL: GSK.SGSKLOAD                                  N
```

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Locations (2 of 2)

   Specify the following for your customization, then press Enter
   to continue.

 Locations of HFS Resident Components

   WebSphere Application Server product directory:
     /usr/lpp/zWebSphere/V6R0
```

4. On the ″Define Variables to Configure Managed Node″ panel, type the appropriate number in the *Option* field to select ″System Environment Customization″ and press **Enter**.
5. Fill in the ″System Environment Customization″ panel using the following screen shot as your guide. When you are done, press **Enter**.

System Environment Customization

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Environment Customization (1 of 1)

   Specify the following to customize your system environment, then
   press Enter to continue.

 WebSphere Application Server for z/OS Configuration HFS Information

   Mount point....:  /WebSphere/V6R0
   Name..........:  OMVS.WAS.CONFIG.HFS
   Volume, or '*' for SMS.:  *
   Primary allocation in cylinders...:  250
   Secondary allocation in cylinders.:  100
```

6. On the ″Define Variables to Configure Managed Node″ panel, type the appropriate number in the *Option* field to select ″Server Customization″ and press **Enter**.
7. Fill in the ″Server Customization″ panels using the following screen shots as your guides. When you are done with each panel, press **Enter**.

Server Customization

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (1 of 4)

   Specify the following to customize your node, then press Enter
   to continue.

 Cell and Node Definitions
```

```
   WebSphere Application Server home directory:
     /WebSphere/V6R0
         / AppServer

   Node Host Name:


   Cell name (short)......:  AQTS
   Cell name (long).......:  AQTS

   Node name (short)......:  AQTS
   Node name (long).......:  AQTS

   Admin asynch operations procedure name:  BBOW6SH
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (2 of 4)

   Specify the following to customize your node, then press Enter
   to continue.

 Procedure Name Definitions

   Controller Information

     Procedure name:  BBO6ACR
     User ID.......:  ASCR1
     UID...........:  2431

   Servant Information

     Procedure name:  BBO6ASR
     User ID.......:  ASSR1
     UID...........:  2432

   Control Region Adjunct

     Procedure name:  BBO6CRA
     User ID.......:  ASCRA1
     UID...........:  2433
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (3 of 4)

   Specify the following to customize daemon definitions for your
   node, then press Enter to continue

 Location Service Daemon Definitions

   Daemon home directory:
     /WebSphere/V6R0/Daemon

   Daemon jobname:  BBODMNB

   Procedure name.:  BBO6DMN
   User ID........:  WSDMNCR1
   UID............:  2411
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (4 of 4)
   Specify the following, which will be used in a job to
   federate your node into the specified Deployment Manager cell.
```

```
WebSphere Application Server home directory:
    /WebSphere/V6R0
        / AppServer
Deployment Manager Access
  Node host name...........:
  JMX SOAP port............:  8879
  Deployment manager security is enabled:  N
     User ID.............:  WSADMIN

Node group name.......:  DefaultNodeGroup

Node Agent Definitions
  Server name (short)...:  BBON001
  Server name (long)....:  nodeagent
  JMX SOAP connector port........:  9360
  Node Discovery port............:  7272
  Node Multicast Discovery port..:  5000

The High Availability Manager Host must not be multihomed
  High Availability Manager Host:
  High availability manager communication port:  9354

ORB listener host name...:  *
  ORB port.......................:  2809
  ORB SSL port...................:  0
```

Note: The ″High Availability Manager Host″ field was removed from the Customization Dialog in WebSphere Application Server for z/OS Version 6.0.2.

8. On the ″Define Variables to Configure Managed Node″ panel, type the appropriate number in the *Option* field to select ″View Security Domain Configuration Panels″ and press **Enter**. These panels display values you previously set in the ″Configure security domain″ option--you cannot change any of the values here. If you do want to make changes, you must go back to the main dialog panel and run through the ″Configure security domain″ option again.

Creating the customization jobs and files

You must select configuration data sets to use and complete the process of defining variables for this task. See “Choosing configuration data sets” on page 68 and “Setting the customization variables: Managed node” on page 69 for more information.

The Customization Dialog creates customization batch jobs and data files, based on the variable values you specified in the dialog. The batch jobs and data sets will be written to the *config_hlq*.CNTL and *config_hlq*.DATA configuration data sets that you created with the ″Allocate target data sets″ option.

1. Ensure that the configuration data sets are allocated and not in use.

   Note: Editing a member in *config_hlq*.CNTL or *config_hlq*.DATA will cause this task to fail.
2. On the ″Configure Managed Node″ panel, type the appropriate number in the *Option* field to select ″Generate customization jobs″ and press **Enter**. You will have one of two results:
   - **Result A:** If all variables are defined correctly, you see the ″Specify Job Cards″ panel, which looks similar to this:

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Generate Customization Jobs

 This portion of the Customization Dialog generates the jobs you must
 run after you complete this Dialog process. You must complete the
 customization process before you generate the jobs with this step.
 If you have not done this, please return to that step.

 Jobs and data files will get generated into data sets:
   'hlq.CNTL'
   'hlq.DATA'
```

```
If you wish to generate customization jobs using other data sets, then
exit from this panel and select the "Allocate target data sets" option.

All the jobs that will be tailored for you will need a job card.
Please enter a valid job card for your installation below. The
file tailoring process will update the jobname for you in all the
generated jobs, so you need not be concerned with that portion of
the job cards below. If continuations are needed, replace the
comment cards with continuations.

Specify the job cards, then press Enter to continue.

//jobname  JOB (ACCTNO,ROOM),'userid',CLASS=A,REGION=0M
//*
//*
//*
```

Note: Pay particular attention to the displayed target data sets. Make sure that they are the ones you intend to use.

- **Result B:** If the variables are not defined correctly, you will see the ″Verification″ panel. Decide whether the warnings or errors are serious enough to warrant returning to the ″Define variables″ option.

  Note: If the return code is 8 or greater, return to the ″Define variables″ option and fix the uncovered problems. If you saved the variables previously, be sure to re-save them after making any updates.

3. Fill in the job card information, according to your installation requirements. For each job, the dialog generates a jobname and the ″JOB″ keyword to match the member name of the PDS, but you specify the rest.

   Note: If you need to run these jobs on a particular system in the sysplex (for example, JES2 MAS or JES3 complex), you should specify the necessary Scheduling Environment (SCHENV), JES2 JOBPARM, or JES3 //*MAIN statement at this time.

   Example of a job card entry:

   ```
   //jobname JOB 1234,USER1,NOTIFY=????,MSGCLASS=O,REGION=0M
   //*             USER=SYSADM1,PASSWORD=SYSADM1
   /*JOBPARM SYSAFF=SYSB
   ```

   Note: This example is useful for jobs that require a user ID other than that of the logged-on TSO user. (This is typically a user ID with UID=0.) In that case, you can just put a comma at the end of the first line, put in the correct user ID on the second line, then uncomment that second line.

   You might want to use RACF SUBMIT authority to avoid having to keep passwords in your configuration data sets.

4. Fix any errors. If there are errors anywhere, you will see the ″Error″ panel. Press PF3 to exit the error panel, then enter the correct panel to fix the errors. Then return to the ″Generate Customization Jobs″ option and pick up where you left off. If necessary, you can update the variables and rerun this option. The generation process will delete and re-tailor all the members.

   Note: Compress the configuration data sets before you rerun this option.

You are done when all the jobs are generated. You can move ahead to viewing the generated jobs. See “Following the generated customization instructions: Managed node” for more information.

Following the generated customization instructions: Managed node

You must generate the customization jobs and files for this task.

The Customization Dialog creates a set of instructions for each customization task. Follow these instructions to tailor and customize a managed node on your system.

Note: Do not attempt to fix a typographical error or make a change by modifying the generated output. Many of the variables are used in multiple members of the target data sets; if you do not change them all, therefore, you will run into problems that are very difficult to diagnose.

1. On the ″Configure Managed Node″ panel, type the appropriate number in the *Option* field to select ″View instructions″ and press **Enter**. ISPF Browse will open and you will see the BBOMNINS member of *config_hlq*.CNTL.
2. Read the instructions carefully, both to preview the customization process and to find any typographical or other errors you might have made while entering the customization variable values.
3. Follow the instructions as given. There are two ways to follow the directions:
   - Follow the instructions while remaining in ISPF Browse.
   - Record the data set name and member at the top of the screen and either print the instructions or use ISPF split screen and browse or edit the instructions while you follow them.
4. Fix any problems. If you encounter problems caused by your Customization Dialog values, modify your variables using the dialog, regenerate the instructions, and restart the customization process.

   Note: Remember that you cannot generate new customization jobs while either configuration data set is open!

You are done with this customization task when you have successfully followed the generated instructions.

Sample generated instructions: Managed node:

This article presents a sample of what the Customization Dialog's generated instructions might look like. This is a sample only--you must use the instructions generated from your own variables when configuring your system.

Note: This sample is based on the installation of WebSphere Application Server for z/OS Version 6.0.2. Generated instructions from different releases of Version 6 would have different content.

```
-----------------------------------------------
Instructions for customizing WebSphere for z/OS
for a Managed node.

The customization dialog has created jobs based on the information you
provided. These instructions tell you how to modify the operating
system and run the jobs to customize WebSphere for z/OS.

RULES:

1.  If you created the target data sets (*.CNTL and *.DATA) on another
    (driving) system, you must copy them to the target system and give
    them the same data set names.

2.  You must perform these instructions on your target system.

Doing manual configuration updates
----------------------------------

The customization dialog for WebSphere for z/OS does not attempt to
update configuration data for your base operating system or existing
subsystems. You must do the following manual steps prior to running
the WebSphere for z/OS configuration jobs.

Perform these steps to do manual configuration updates:

1.  Update BLSCUSER. Refer to member BBOIPCSP in

    'USERID.TODAY.CNTL'

    In order to use the IPCS support provided by the product, append
    the contents of this member to the BLSCUSER member in your IPCSPARM
    or system PARMLIB datasets.

    -------------------------------------------------------------------

2.  Update SCHEDxx. Refer to member BBOSCHED in

    'USERID.TODAY.CNTL'

    In order to set the correct program properties for the WebSphere
    for z/OS run-time executables, append the contents of this member
    to the SCHEDxx member in your system PARMLIB concatenation.

    Note: When you are finished, issue the command SET SCH=(xx,xx)
    to activate SCHEDxx and load a new program properties table.

    -------------------------------------------------------------------

3.  Make sure the following data sets are APF-authorized:

        DBEL.WASX.SBBOLPA
        DBEL.WASX.SBBOLOAD
        DBEL.WASX.SBBGLOAD
        DBEL.WASX.SBBOLD2

    Add these datasets to your PROGxx or IEAAPFxx parmlib members, as
    appropriate, ensuring you specify the correct volsers.

    -------------------------------------------------------------------

4.  If you want to collect the SMF120 records created by the run-time
    servers, update SMFPRMxx via the following:

    EXAMPLE:

       SUBSYS(STC,EXITS(IEFU29,IEFACTRT),INTERVAL(SMF,SYNC),
                          TYPE(0,30,70:79,88,89,120,245))
                                                ---

    For details on the SMF records, see related topics in the
    WebSphere for z/OS Information Center at
    http://www.ibm.com/software/webservers/appserv/zos_os390/library/

    -------------------------------------------------------------------

5.  Update your active BPXPRMxx member to have the following WebSphere
    for z/OS configuration file system:

    OMVS.WAS.CONFIG.HFS

    mounted at:

    /WebSphere/V6R1

    in read/write mode.

    EXAMPLE:

       MOUNT FILESYSTEM('OMVS.WAS.CONFIG.HFS')
         MOUNTPOINT('/WebSphere/V6R1')
          TYPE(HFS)
          MODE(RDWR)

    -------------------------------------------------------------------

6.  Update TCP/IP by reserving the following ports for WebSphere for
    z/OS.  These will be used during the federation process of your
    managed node.
```

```
       SOAP JMX Connector port                      - 9360
       Node Discovery port                          - 7272
       Node Multicast Discovery Port                - 5000
       Node Agent's ORB port                        - 2809
       High Availability Manager Communication port - 9354

    View member BBOTCPIM in

    'USERID.TODAY.CNTL'

    Add the contents of this member to the PORT section of the file
    referenced by the DD statement for the TCP/IP profile in the
    TCP/IP start procedure. Cut and paste from this member into the
    data set used by your installation.

    ATTENTION: If another application has already reserved any of these
    ports for its own use, you must resolve the resulting conflict
    before you continue. If you update the WebSphere for z/OS
    customization dialog with new port specifications, be sure to
    regenerate the customization jobs, data, and instructions.


Note: The addNode process introduces a special utility server to
      the node.  This utility server is called a nodeagent and
      exists to support administrative functions on the node. By
      default the nodeagent takes over ORB port 2809.   Note on
      WebSphere z/OS the ORB port doubles as the INS CosNaming
      bootstrap port.  By default, this port (2809) was assigned
      to the Application Server.  Normally you want the nodeagent
      to be the INS CosNaming bootstrap point for the entire node,
      so that RMI/IIOP clients that do not override the INS
      CosNaming bootstrap defaults can locate within the namespace,
      EJBs installed on any server on that node.  In order for the
      nodeagent to take over port 2809,  the Application Server must
      be assigned a new ORB port.  The default new ORB port for the
      Application Server is 9810.  The nodeagent will take over a
      Application Server's ORB port if and only if the nodeagent's ORB
      port is equal to an Application Server's ORB port.  You can
      specify the nodeagent's ORB port in the 'ORB port' field.
      You can specify the new ORB port for the Application Server in
      the 'Appplication Server's ORB Port' field.

    ATTENTION: Skip this step if the ports are already defined in the
    TCP/IP profile.

    --------------------------------------------------------------------

7.  The WebSphere product libraries will be placed in STEPLIB as
    needed, rather than in the system link pack area and system link
    list. Make sure that the target MVS system has at least 8MB
    of free storage in extended CSA for the daemon and for EACH
    node (deployment manager node or application server node).

    SBBOLOAD and SBBOLD2:
    ====================
    The following data sets will be placed in the STEPLIB concatenation
    for the location service daemon, controller and servant regions,
    and in the setupCmdLine.sh script in the WebSphere Configuration
    file system. You must not remove these STEPLIB statements.

    DBEL.WASX.SBBOLOAD
    DBEL.WASX.SBBOLD2


    BBORTS61:
    =========
    The BBORTS61 module is used by WebSphere Application Server for
    component trace support. A copy of this module (any maintenance
    level) must be in the system link pack area in order for CTRACE
    to work correctly.

    If a copy of BBORTS61 is currently loaded into LPA, you need take
    no further action.

    Otherwise, issue the following MVS console command to load BBORTS61
    into dynamic LPA:

      SETPROG LPA,ADD,MODNAME=BBORTS61,
              DSNAME=DBEL.WASX.SBBOLPA

    Alternatively, you can place the following statement in a parmlib
    PROGxx member which is activated with the SET PROG= command after
    system IPL is complete:

    LPA ADD MODNAME(BBORTS61)
      DSNAME(DBEL.WASX.SBBOLPA)

    Make sure that the BBORTS61 module is loaded into LPA after each
    system IPL.

    -------------------------------------------------------------------

8.  WebSphere for z/OS customization assumes that the following system
    data sets are in the system link list or link pack area:

    Language Environment     SCEERUN
                             SCEERUN2

    System SSL               SGSKLOAD (z/OS 1.5 and below)
                             SIEALNKE (z/OS 1.6 and above)

    Placing these data sets in the link list or link pack area improves
    performance and insulates your WebSphere for z/OS configuration
    from changes in data set names (for example, when migrating to z/OS
    1.6).

    If the Language Environment or System SSL load module libraries are
    not in your system link list or link pack area, you must perform
    the following steps before starting any WebSphere Application
    Server for z/OS servers:

    - Make sure the data sets are APF-authorized
    - Complete the optional step below to add the data sets to STEPLIB
      in the server JCL and setupCmdLine.sh script(s).

    If you regenerate server cataloged procedures at any point, make
    sure the data sets are added to the new cataloged procedures.

    -------------------------------------------------------------------

9.  If the error logstream
    WAS.ERROR.LOG
    does not already exist on your target system, make a copy of the
    appropriate job in the SBBOJCL data set, customize it according
    to the comments in the job, and run it:

       BBOERRLC    Create an error logstream in a coupling facility
       BBOERRLD    Create a DASD-only error logstream

    -------------------------------------------------------------------

10. WebSphere for z/OS regions open a large number of files (more than
    1024). Make sure your BPXPRMxx parmlib member(s) specify a value of
    MAXFILEPROC that is greater than or equal to 2000. Use the
    following MVS console command to see your current MAXFILEPROC
    setting:

      D OMVS,OPTIONS

Running the customized jobs
---------------------------

The customization dialog built a number of batch jobs with the
variables you supplied. You must run the jobs in the order listed
below using user IDs with the appropriate authority.

BEFORE YOU BEGIN: Complete the section above entitled "Doing manual
configuration updates."

Follow the table below, which lists in order the jobs you must submit
and the commands you must enter. Special handling notes are included
in the table. All jobs are members of

USERID.TODAY.CNTL

Attention: After submitting each job, carefully check the output.
Errors may exist even when all return codes are zero.

+-----------+--------------------------------------------------------+
| BBOMSGC   | User ID requirement: Update authority for data set     |
+-----------+ SYS1.MSGENU and/or SYS1.MSGJPN.                        |
| Done:     |                                                        |
|           | ATTENTION: This is optional unless you require message |
|           | translation.                                           |
| By:       |                                                        |
|           | This job sets up MMS to translate messages for WebSphere|
|           | for z/OS.                                              |
|           |                                                        |
|           | There are two steps to update SYS1.MSGENU and          |
|           | SYS1.MSGJPN. Remove the unneeded step and change the   |
|           | target libraries, if necessary.                        |
+-----------+--------------------------------------------------------+
|           | Check WebSphere Application Server home directories.   |
+-----------+                                                        |
| Done:     | Verify that the following directories exist on your    |
|           | target z/OS system and that the ownership and permission|
|           | bits are correct:                                      |
| By:       |                                                        |
|           | /var/WebSphere/home                                    |
|           | ownership: (any)                                       |
|           | permission bits: 755                                   |
|           |                                                        |
|           | /var/WebSphere/home/WSCFG1                             |
|           | ownership: WSOWNER:WSCFG1                              |
|           | permission bits: 770                                   |
|           |                                                        |
|           |                                                        |
|           | /var/WebSphere/home/WSSR1                              |
|           | ownership: WSOWNER:WSSR1                               |
|           | permission bits: 770                                   |
|           |                                                        |
|           |                                                        |
|           | /var/WebSphere/home/WSCLGP                             |
|           | ownership: WSOWNER:WSCLGP                              |
|           | permission bits: 770                                   |
|           |                                                        |
|           | If the these directories do not exist, create them with|
|           | the above ownership and permission bits.               |
|           |                                                        |
|           | The security domain configuration job BBOSBRAM can be    |
|           | used to create these directories if necessary.           |
|           |                                                          |
+-----------+----------------------------------------------------------+
| BBOMBRAJ  | User ID requirement: Authority to update data set        |
+-----------+                                                          |
| Done:     | USERID.TODAY.DATA                                        |
|           |                                                          |
|           | This job builds (but does not execute) the RACF commands |
| By:       | for the WebSphere for z/OS run-time clusters and places  |
|           | them into member BBOMBRAK of data set                    |
|           |                                                          |
|           | USERID.TODAY.DATA                                        |
|           |                                                          |
|           | Carefully review these definitions with your security    |
|           | administrator.                                           |
+-----------+----------------------------------------------------------+
| BBOMBRAK  | User ID requirement: RACF special authority.             |
+-----------+                                                          |
| Done:     | This job executes the RACF commands set up in the        |
|           | previous job.                                            |
|           |                                                          |
|           | RESULT: You may receive errors, such as INVALID USER     |
| By:       | messages, from this job because a user ID, group  or     |
|           | profile is already defined.  Make sure the existing      |
|           | user ID, group or profile has the same characteristics   |
|           | as the user ID, group or profile being created by        |
|           | BBOMBRAK.  If not, then change the values in the         |
|           | customization dialog which are causing the conflict,     |
|           | regenerate the customization jobs, and restart the       |
|           | process.                                                 |
|           |                                                          |
+-----------+----------------------------------------------------------+
| --------  | Check user ID authorizations.                            |
+-----------+                                                          |
| Done:     | Make sure the WSCFG1 group has read access to all        |
|           | WebSphere product data sets, as well as to any other     |
|           | data sets which will be placed in WebSphere for z/OS     |
|           | cataloged procedure STEPLIB concatenations.              |
|           |                                                          |
| By:       | the resolver configuration file in use on your system.   |
|           | Depending on your IP setup, this file may be             |
|           | /etc/resolv.conf, SYS1.TCPPARMS(TCPDATA), or another     |
|           | data set.                                                |
|           |                                                          |
|           | ASCR1                                                    |
|           | ASSR1                                                    |
|           |                                                          |
|           | See the z/OS eNetwork Communication Server IP            |
|           | Configuration manual for the resolver search order.      |
|           |                                                          |
|           | Ensure the following user ID has read access to the data |
|           | sets in your system parmlib concatenation:               |
|           |                                                          |
|           | WSDMNCR1                                                 |
|           |                                                          |
|           | ATTENTION:                                               |
|           |                                                          |
|           |  If operator commands are protected by the z/OS security |
|           |  server at your installation, you must ensure that       |
|           |  sufficient authority is given to WebSphere tasks to     |
|           |  control operations.                                     |
|           |                                                          |
|           |  The Application Server Controller user ID (ASCR1)       |
|           |  needs the ability to perform operations on started      |
|           |  tasks belonging to WebSphere Application Server for     |
|           |  z/OS.                                                   |
|           |                                                           |
|           |  The asynchronous administrator user ID, and any user     |
|           |  ID used to run the federation job when the node agent    |
|           |  is started automatically, need the authority to issue    |
|           |  the MVS START command.                                   |
|           |                                                           |
|           |  If you are currently controlling MVS console command     |
|           |  authority with SAF OPERCMDS profiles, grant the          |
|           |  following authorities as indicated, substituting your    |
|           |  own profile names:                                       |
|           |                                                           |
|           |  PERMIT  START_profile_name  CLASS(OPERCMDS)              |
|           |          ID (ASCR1  WSADMSH)  ACCESS(UPDATE)              |
|           |                                                           |
|           |  PERMIT  STOP_profile_name  CLASS(OPERCMDS)               |
|           |          ID (ASCR1 )  ACCESS(UPDATE)                      |
|           |                                                           |
|           |  PERMIT  MODIFY_profile_name  CLASS(OPERCMDS)             |
|           |          ID (ASCR1 )  ACCESS(UPDATE)                      |
|           |                                                           |
|           |  PERMIT  CANCEL_profile_name  CLASS(OPERCMDS)             |
|           |          ID (ASCR1 )  ACCESS(UPDATE)                      |
|           |                                                           |
|           |  PERMIT  FORCE_profile_name  CLASS(OPERCMDS)              |
|           |          ID (ASCR1 )  ACCESS(UPDATE)                      |
|           |                                                           |
|           |  You must also grant the appropriate console command      |
|           |  authority to any user ID that executes the               |
|           |  startServer.sh or stopServer.sh script.                  |
|           |                                                           |
+-----------+-----------------------------------------------------------+
| BBOMCHFS  | User ID requirement: UID=0 and authority to allocate      |
+-----------+                                                           |
| Done:     | OMVS.WAS.CONFIG.HFS                                       |
|           |                                                           |
|           | This job:                                                 |
|           |                                                           |
| By:       | o   Creates a mount point directory                       |
|           |                                                           |
|           |     /WebSphere/V6R1                                       |
|           |                                                           |
|           | o   Allocates the configuration file system using         |
|           |     the Hierarchical File System (HFS)                    |
|           |                                                           |
|           |     OMVS.WAS.CONFIG.HFS                                   |
|           |                                                           |
|           |     and mounts it at the above mount point.               |
|           |                                                           |
|           | DO NOT RUN THIS JOB IF:                                   |
|           |   1. The configuration file system already exists and is  |
|           |      mounted at the desired mountpoint, or if             |
|           |                                                           |
|           |   2. The mount point directory is controlled by           |
|           |      automount.  Either disable the automount rule for    |
|           |      the configuration mount point while running this     |
|           |      job, or perform the following steps manually:        |
|           |                                                           |
|           |      a. Allocate the configuration file system data set.  |
|           |      b. Issue the following shell commands, which will    |
|           |         also cause automount to mount the file system     |
|           |                                                           |
|           |      chmod 775 /WebSphere/V6R1                            |
|           |      chown WSOWNER:WSCFG1 /WebSphere/V6R1                 |
|           |                                                           |
|           |                                                           |
|           | BEFORE YOU BEGIN: The BBOMCHFS job assumes your root      |
|           | file system is mounted in read/write mode.  If the root   |
|           | file system is not mounted in read/write mode, manually |
|           | create the directory                                    |
|           |                                                         |
|           | /WebSphere/V6R1                                         |
|           |                                                         |
|           | and any needed higher directories, set file permissions |
|           | to 775, and set the owning user ID and group to WSOWNER |
|           | and WSCFG1 before running BBOMCHFS.                     |
|           |                                                         |
|           | EXAMPLE: If you plan to use /WebSphere/V6R0M0 as your   |
|           | directory, issue the following commands from within the |
|           | OMVS shell:                                             |
|           |                                                         |
|           |   mkdir -p -m 775 /WebSphere/V6R0M0                     |
|           |   chown -R WSOWNER:WSCFG1 /WebSphere                    |
|           |                                                         |
+-----------+---------------------------------------------------------+
| BBOMHFSA  | User ID requirement: UID=0.                             |
+-----------+                                                         |
| Done:     | This job populates the previously-created file system.  |
|           |                                                         |
|           | Upon completion, examine the job output. Success is     |
| By:       | indicated with a RC=0 in the job output.                |
|           |                                                         |
+-----------+---------------------------------------------------------+
| BBOWCPYM  | User ID requirement: update authority for:              |
+-----------+                                                         |
| Done:     |     SYS1.PROCLIB                                        |
|           |                                                         |
| By:       |                                                         |
|           |                                                         |
|           |                                                         |
|           | This job copies the tailored start procedures,          |
|           | parameters, and EXECs to the run-time libraries.        |
|           |                                                         |
|           | ATTENTION: Be aware that you may overlay existing       |
|           | members in the above data sets.                         |
|           |                                                         |
+-----------+---------------------------------------------------------+
| BBOWWPFM  | User ID requirement: UID=0.                             |
+-----------+                                                         |
| Done:     | This job sets up the runtime file system.               |
|           |                                                         |
|           |                                                         |
| By:       | Upon completion, examine the job output. Success is     |
|           | indicated by rc=0.                                      |
|           |                                                         |
|           | Note: If the BBOWWPFM (profile creation) job fails,     |
|           | delete the WAS_HOME/profiles/default directory and all  |
|           | its contents before rerunning BBOWWPFM.                 |
|           |                                                         |
+-----------+---------------------------------------------------------+
| BBOMHFSB  | User ID requirement: UID=0.                             |
+-----------+                                                         |
| Done:     | This job will complete the file system initialization.  |
|           |                                                         |
|           | Upon completion, examine the job output. Success is     |
| By:       | indicated with a RC=0 in the job output.                |
|           |                                                         |
+-----------+---------------------------------------------------------+
| --------  | All WebSphere Application Server processes require      |
+-----------+ access to the Language Environment and System SSL load  |
| Done:     | modules.                                                |
|           |                                                         |
|           | If the SCEERUN, SCEERUN2 and System SSL load module     |
|           | libraries are not in the system link list or link pack  |
|           | area, add them to the STEPLIB DD concatenation in each  |
|           | of the following cataloged procedures in                 |
|           | SYS1.PROCLIB:                                            |
|           |                                                          |
|           |     BBO6ACRZ                                             |
|           |     BBO6ASRZ                                             |
|           |     BBO6CRAZ                                             |
|           |     BBO6DMNZ                                             |
|           |                                                          |
|           | and also add the full data set names, separated by       |
|           | colons (:), to the STEPLIB variable in the shell script  |
|           |                                                          |
|           |     /WebSphere/V6R1/                                     |
|           |      AppServer/                                          |
|           |       profiles/default/bin/setupCmdLine.sh               |
| By:       |                                                          |
|           | When modifying the setupCmdLine.sh script, do not        |
|           | remove lines or comment them out, as this may cause      |
|           | problems with automated updates to the script.           |
|           |                                                          |
|           | Add only those data sets which are NOT in the link list  |
|           | or link pack area.                                       |
|           |                                                          |
+-----------+----------------------------------------------------------+
| --------  | Make sure Resource Recovery Services (RRS) is active.    |
+-----------+ (See the InfoCenter for setup instructions if necessary.)|
| Done:     | Look for the following console message to verify that    |
|           | RRS was successfully started:                            |
|           |                                                          |
| By:       |                                                          |
|           |   ASA2011I RRS INITIALIZATION COMPLETE. COMPONENT        |
|           |     ID=SCRRS                                             |
|           |                                                          |
+-----------+----------------------------------------------------------+
| BBOWMNAN  | User ID requirement: WSADMIN                             |
+-----------+                                                          |
| Done:     | If you choose the start the node agent automatically,    |
|           | the user ID used to run BBOWMNAN will also need the      |
|           | authority to issue the MVS START command.                |
|           |                                                          |
|           | This job will federate your node into the specified      |
|           | Deployment Manager cell.  Ensure that your Deployment    |
|           | Manager is running before submitting this job            |
|           |                                                          |
|           | Upon completion, examine the job output. Success is      |
| By:       | indicated with a RC=0 in the job output.                 |
|           |                                                          |
+-----------+----------------------------------------------------------+
| --------  | If your system is busy, you may want to include a rule   |
+-----------+ in your WLM policy that OMVS work for job BBON001        |
| Done:     | (such as the postinstaller step) is to run in a service  |
|           | class with a high service objective.                     |
|           |                                                          |
| By:       |                                                          |
+----------------------------------------------------------------------+
| --------  | Start the node agent server                              |
+-----------+                                                          |
|           |                                                          |
|           | Note: The node agent is automatically started by the node|
|           | federation process.  This step is information, for       |
|           | starting the node agent at other times.                  |
|           |                                                          |
|           | Issue the following MVS command to start your node agent |
|           | server, replacing dmgr_cell_short_name with the     |
| Done:     | cell short name of the target Deployment Manager cell    |
|           |                                                          |
| By:       |   START BBO6ACR,JOBNAME=BBON001,                         |
|           |         ENV=dmg_cell_short_name.AQFT.BBON001        |
|           |                                                          |
|           | RESULT: The following message appears on the console and |
|           | in the job log of BBON001.                              |
|           |   BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR   |
|           |   z/OS CONTROL PROCESS BBON001                          |
|           |                                                         |
+-----------+---------------------------------------------------------+
+-----------+---------------------------------------------------------+
| The product is now configured.  You may create and manage           |
| application servers in the node using the administrative console or |
| scripting.                                                          |
|                                                                     |
+---------------------------------------------------------------------+
```

Working with your new managed server node

Once you complete the customization instructions, you will have a WebSphere Application Server for z/OS application server node (managed node) in your Network Deployment cell.

Before starting your server

Make sure that the WebSphere Application Server for z/OS product HFS and configuration HFS are mounted. If you chose to load the SBBOLPA (and possibly SBBOLOAD) into the system link pack area, make sure that these libraries are loaded into LPA before starting the server.

Starting the node agent

To start your node agent, issue the following MVS console command:

```
START server_proc,JOBNAME=nodeagent_name,ENV=cell_name.node_name.nodeagent_name
```

where

- *server_proc* is the node agent cataloged procedure.
- *nodeagent_name* is the node agent short name.
- *node_name* is the node short name.
- *cell_name* is the cell short name.

If you chose default values for example (your sysplex is named CELL1 and your system is named MVSA), you would enter the following START command:

```
START BBO6ACR,JOBNAME=BBON001,ENV=CELL1.MVSA.BBON001
```

The START command brings up the node agent. The node agent starts the location service daemon (if one is not already running). You should see a message like the following when the node is up and running:

```
BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR z/OS CONTROL PROCESS BBON001
```

The node agent must be running in order for the deployment manager to administer the node.

Administering the node through the administrative console

If the deployment manager for the cell is up and running, access the administrative console by pointing a Web browser to the following URL:

```
http://hostname:http_port/ibm/console
```

where

- *hostname* is the deployment manager HTTP transport host name you specified during customization.

  Note: If you specified "*" for the deployment manager HTTP host name, this is actually the deployment manager node host name.

- *http_port* is the deployment manager HTTP port you specified during customization.

  Note: The default HTTP port for the deployment manager is 9060.

Until administrative security is enabled, you will see a signon screen that asks you for a user ID but no password.

The user ID can be anything and is used only to provide basic tracking of changes. Be aware that until you enable administrative security, anyone with a Web browser and access to the HTTP port can modify your application serving environment.

You can use the administrative console, scripting, or both to manage the node and deploy and manage J2EE applications. See Using the administrative clients for more information. Before you can deploy applications, however, you need to add application servers to your managed node.

Adding application servers

Application servers can be added to the managed server node using the administrative console or scripting. Either of two methods can be used:

- Create a new application server directly using the administrative console or scripting. See “Creating application servers” on page 251 for more information. You can use the controller, servant and CRA cataloged procedures and user IDs created during the managed node setup process for any application servers you create in the managed node.
- Cluster an existing application server in another node, using this managed node as a target. This will create a ″cloned″ copy of the application server being clustered in your new managed node. See “Creating clusters” on page 397 for more information.

Starting application servers

To start one of your managed node’s application servers, issue the following MVS console command:

```
START server_proc,JOBNAME=server_name,ENV=cell_name.node_name.server_name
```

where

- *server_proc* is the application server agent cataloged procedure (can be the same as the node agent cataloged procedure).
- *server_name* is the application server short name.
- *node_name* is the node short name.
- *cell_name* is the cell short name.

If you chose the default procedure name for example (your sysplex is named CELL1, your node is named MVSA, and your server is named AZSR01A), you would enter the following START command:

```
START BBO6ACR,JOBNAME=AZSR01A,ENV=CELL1.MVSA.AZSR01A
```

The START command brings up the application server controller. The controller starts the location service daemon (if one is not already running) and then uses WLM to start the control region adjunct and the servants. You should see a message similar to the following when the node is up and running:

```
BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR Z/OS CONTROL PROCESS AZSR01A
```

Stopping your managed node

Use one of the following two methods to stop your deployment manager:

- Stop the location service daemon, which also stops any of the cell’s nodes on the same z/OS system. The location service daemon holds pointers to modules in common storage, and stopping it forces all cell members on the same z/OS system as the daemon to shut down. To stop the location service daemon, enter the following MVS console command:

```
STOP daemon_jobname
```

where *daemon_jobname* is the location service daemon jobname. The default location service daemon jobname for a Network Deployment cell is BBODMNC.

- Stop just the node agent and its application servers while leaving the location service daemon, deployment manager (if present), and any other managed nodes on the z/OS system still running. To stop the node agent, enter the following MVS console command:

  ```
  STOP nodeagent_name
  ```

  where *nodeagent_name* is the node agent short name. The default node agent short name is BBON001.

Federating a stand-alone application server into a Network Deployment cell

This article leads you through the tasks involved in federating a WebSphere Application Server for z/OS stand-alone application server into a Network Deployment cell.

Perform this task to federate an existing WebSphere Application Server for z/OS stand-alone application server into a Network Deployment cell.

1. Log on to TSO on the z/OS system on which you intend to federate the server. Use a user ID that has READ access to the WebSphere Application Server for z/OS product data sets. You will also need access to a user ID with authority to make security system updates and a user ID with UID 0. (These can all be the same user ID.)
2. Start the Customization Dialog. See “Starting the Customization Dialog” on page 3 for details.
3. Choose the configuration data sets in which you will store your customization jobs and data. See “Choosing configuration data sets” on page 87 for details.
4. Load the security domain variables saved from the security domain you intend to use for this cell. See “Loading the common MVS groups and users variables” on page 86 for details.
5. Set the customization variables according to the values recorded on your federated application server node worksheet. See “Setting the customization variables: Federated application server node” on page 88 for details.
6. (Optional but recommended.) Save the federated application server node customization variables in a data set. See “Saving the cell variables” on page 52 for details.
7. Create the customization jobs and files, based on the customization variable values you entered. See “Creating the customization jobs and files” on page 89 for details.
8. Follow the generated customization instructions. See “Following the generated customization instructions: Federated application server node” on page 90 for details, and a sample set of customization instructions.

You are done when you have successfully completed the steps in the generated instructions. The new federated application server node is up and running on the chosen z/OS system. See “Working with your new federated server node” on page 93 for more information.

Loading the security domain variables

This article describes how to complete the ″Load security domain variables″ option for a WebSphere Application Server for z/OS federated application server node.

Create the security domain you will use for the new federated application server node and know the name of the saved security domain configuration variable file that you recorded on the security domain worksheet.

The security domain settings are used in the customization of every WebSphere Application Server for z/OS cell. By loading the security domain variables at the start of node or cell creation, you ensure that the security domain configuration you use is consistent and matches the RACF definitions that have already been set as part of security domain configuration.

Complete this task as the first step in configuring a new federated application server node. If you encounter problems during customization and change the security domain variable values, be sure to re-save them.

1. On the ″Federate stand-alone application server node″ panel, type the appropriate number in the *Option* field to select ″Load security domain variables″ and press **Enter**. You will see a panel that looks similar to the following:

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Load Security Domain Variables

 Specify the name of a data set containing the security domain variables,
 then press Enter to continue.

 IBM-supplied defaults are in ''


 Data set name:


 If this data set is not cataloged, specify the volume.

 Volume:
```

2. Fill in the name of the sequential data set you used to hold the security domain variable values and press **Enter**. The security domain variables will load.

The security domain settings are loaded. You can display these variables, but not change them.

Loading the common MVS groups and users variables

This article describes how to complete the ″Load common MVS groups and users variables″ option for a WebSphere Application Server for z/OS federated application server node.

Create the common MVS groups and users you will use for the new federated application server node and know the name of the saved common MVS groups and users configuration variable file that you recorded on the common MVS groups and users worksheet.

The common MVS groups and users settings are used in the customization of every WebSphere Application Server for z/OS cell. By loading the common MVS groups and users variables at the start of node or cell creation, you ensure that the common MVS groups and users configuration you use is consistent and matches the RACF definitions that have already been set as part of common MVS groups and users configuration.

Complete this task as the first step in configuring a new federated application server node. If you encounter problems during customization and change the common MVS groups and users variable values, be sure to re-save them.

1. On the ″Federate stand-alone application server node″ panel, type the appropriate number in the *Option* field to select ″Load common MVS groups and users variables″ and press **Enter**. You will see a panel that looks similar to the following:

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Load common MVS groups and users Variables
```

```
 Specify the name of a data set containing the common MVS groups and users variables,
 then press Enter to continue.

 IBM-supplied defaults are in ''


 Data set name:


 If this data set is not cataloged, specify the volume.

 Volume:
```

2. Fill in the name of the sequential data set you used to hold the common MVS groups and users variable values and press **Enter**. The common MVS groups and users variables will load.

The common MVS groups and users settings are loaded. You can display these variables, but not change them.

Choosing configuration data sets

This article leads you through the ″Allocate target data sets″ option in the Customization Dialog.

You must start the Customization Dialog and select the ″Federate an existing stand-alone application server node into an existing Network Deployment cell″ option.

Each option in the Customization Dialog saves customization jobs and files in a pair of customization data sets. While is it possible to reuse these data sets, it is safest to create separate data sets for each WebSphere Application Server for z/OS configuration. The best practice is to use the customization data set name prefix (sometimes referred to as ″*config_hlq*″) to indicate the version and release of WebSphere Application Server for z/OS, the task you are performing, and the cell (and, in some cases, the node name) you are configuring.

Complete this task before generating the customization jobs and files.

1. On the main dialog panel, type the appropriate number in the *Option* field to select ″Allocate target data sets″.
2. Press Enter. **Result:** You see a panel that looks similar to the following:

```
-----------------     WebSphere Application Server for z/OS Customization      ------------------
Option  ===>

 Allocate Target Data Sets

 Specify a high level qualifier (HLQ) and press Enter to allocate the
 data sets to contain the generated jobs and instructions. You can
 specify multiple qualifiers (up to 39 characters).

 High level qualifier:                                         .CNTL
                                                               .DATA

 The Dialog will display data set allocation panels. You can make
 changes to the default allocations, however you should not change
 the DCB characteristics of the data sets.

    .CNTL  - a PDS with fixed block 80-byte records to
             contain customization jobs.

    .DATA  - a PDS with variable length data to contain
             other data produced by the Customization Dialog.
```

3. Fill in your chosen configuration data set name prefix value (*config_hlq*). If the data sets *config_hlq*.CNTL and *config_hlq*.DATA do not exist, you will be prompted for data set allocation information. If the data sets already exist, a message will inform you that they will be reused.

The data sets *config_hlq*.CNTL and *config_hlq*.DATA are allocated and will store customization jobs and files. These data set names will also be saved along with the customization variables.

Setting the customization variables: Federated application server node

This article describes how to complete the ″Define variables″ option for a WebSphere Application Server for z/OS federated node.

You must start the Customization Dialog and select the ″Create stand-alone Application Server nodes″ option. Have the Federating an application server node Customization Dialog worksheet completed and at hand. A copy of this worksheet is available in the *Installing your application serving environment* PDF.

1. On the ″Federate stand-alone application server Node″ panel, type the appropriate number in the *Option* field to select ″Define variables″ and press **Enter**.
2. On the ″Define Variables for Federate stand-alone Application Server Node″ panel, type the appropriate number in the *Option* field to select ″Define Variables for Node Federation″ and press **Enter**.
3. Fill in the ″Define Variables for Federate stand-alone Application Server Node″ panels using the following screen shots as your guides. When you are done with each, press **Enter**.

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Define Variables for Federate stand-alone application server Node (1 of 2)
   Specify the following to customize your server, then press Enter
   to continue.

 WebSphere Application Server home directory:
     /WebSphere/V6R0
         / AppServer
 Deployment Manager Access
   Node host name...........:
   JMX SOAP port............:  8879
   Deployment manager security is enabled:  N
      User ID.............:  WSADMIN

 Include Apps..........:  Y
 Launch the node agent after node federation:  Y
 Application server's ORB port..:  9810
 Node group name.......:  DefaultNodeGroup

 Node Agent Definitions
   Server name (short)...:  BBON001
   Server name (long)....:  nodeagent
   JMX SOAP connector port........:  9360
   Node Discovery port............:  7272
   Node Multicast Discovery port..:  5000
   High availability manager communication port:  9354

 ORB listener host name...:  *
   ORB port.......................:  2809
   ORB SSL port...................:  0
```

Note: The ″Launch the node agent after node federation″ field was added to the Customization Dialog in WebSphere Application Server for z/OS Version 6.0.2.

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Define Variables for Federate stand-alone application server Node (2 of 2)
   Specify the following to customize your server, then press Enter
   to continue.

 Do you wish to federate service integration busses
   that exist on this node?" (Y/N):  N
```

Creating the customization jobs and files

You must select configuration data sets to use and complete the process of defining variables for this task. See “Choosing configuration data sets” on page 87 and “Setting the customization variables: Federated application server node” on page 88 for more information.

The Customization Dialog creates customization batch jobs and data files, based on the variable values you specified in the dialog. The batch jobs and data sets will be written to the *config_hlq*.CNTL and *config_hlq*.DATA configuration data sets that you created with the ″Allocate target data sets″ option.

1. Ensure that the configuration data sets are allocated and not in use.

   Note: Editing a member in *config_hlq*.CNTL or *config_hlq*.DATA will cause this task to fail.
2. On the ″Federate stand-alone application server Node″ panel, type the appropriate number in the *Option* field to select ″Generate customization jobs″ and press **Enter**. You will have one of two results:
   - **Result A:** If all variables are defined correctly, you see the ″Specify Job Cards″ panel, which looks similar to this:

     ```
     ------------  WebSphere Application Server for z/OS Customization     --------
     Option  ===>

     Generate Customization Jobs

      This portion of the Customization Dialog generates the jobs you must
      run after you complete this Dialog process. You must complete the
      customization process before you generate the jobs with this step.
      If you have not done this, please return to that step.

      Jobs and data files will get generated into data sets:
        'hlq.CNTL'
        'hlq.DATA'
      If you wish to generate customization jobs using other data sets, then
      exit from this panel and select the "Allocate target data sets" option.

      All the jobs that will be tailored for you will need a job card.
      Please enter a valid job card for your installation below. The
      file tailoring process will update the jobname for you in all the
      generated jobs, so you need not be concerned with that portion of
      the job cards below. If continuations are needed, replace the
      comment cards with continuations.

      Specify the job cards, then press Enter to continue.

      //jobname  JOB (ACCTNO,ROOM),'userid',CLASS=A,REGION=0M
      //*
      //*
      //*
     ```

     Note: Pay particular attention to the displayed target data sets. Make sure that they are the ones you intend to use.
   - **Result B:** If the variables are not defined correctly, you will see the ″Verification″ panel. Decide whether the warnings or errors are serious enough to warrant returning to the ″Define variables″ option.

     Note: If the return code is 8 or greater, return to the ″Define variables″ option and fix the uncovered problems. If you saved the variables previously, be sure to re-save them after making any updates.
3. Fill in the job card information, according to your installation requirements. For each job, the dialog generates a jobname and the ″JOB″ keyword to match the member name of the PDS, but you specify the rest.

Note: If you need to run these jobs on a particular system in the sysplex (for example, JES2 MAS or JES3 complex), you should specify the necessary Scheduling Environment (SCHENV), JES2 JOBPARM, or JES3 //*MAIN statement at this time.

Example of a job card entry:

```
//jobname JOB 1234,USER1,NOTIFY=????,MSGCLASS=O,REGION=0M
//*             USER=SYSADM1,PASSWORD=SYSADM1
/*JOBPARM SYSAFF=SYSB
```

Note: This example is useful for jobs that require a user ID other than that of the logged-on TSO user. (This is typically a user ID with UID=0.) In that case, you can just put a comma at the end of the first line, put in the correct user ID on the second line, then uncomment that second line.

You might want to use RACF SUBMIT authority to avoid having to keep passwords in your configuration data sets.

4. Fix any errors. If there are errors anywhere, you will see the ″Error″ panel. Press PF3 to exit the error panel, then enter the correct panel to fix the errors. Then return to the ″Generate Customization Jobs″ option and pick up where you left off. If necessary, you can update the variables and rerun this option. The generation process will delete and re-tailor all the members.

   Note: Compress the configuration data sets before you rerun this option.

You are done when all the jobs are generated. You can move ahead to viewing the generated jobs. See “Following the generated customization instructions: Federated application server node” for more information.

Following the generated customization instructions: Federated application server node

You must generate the customization jobs and files for this task.

The Customization Dialog creates a set of instructions for each customization task. Follow these instructions to tailor and customize a federated node on your system.

Note: Do not attempt to fix a typographical error or make a change by modifying the generated output. Many of the variables are used in multiple members of the target data sets; if you do not change them all, therefore, you will run into problems that are very difficult to diagnose.

1. On the ″Federate stand-alone application server Node″ panel, type the appropriate number in the *Option* field to select ″View instructions″ and press **Enter**. ISPF Browse will open and you will see the BBOANINS member of *config_hlq*.CNTL.
2. Read the instructions carefully, both to preview the customization process and to find any typographical or other errors you might have made while entering the customization variable values.
3. Follow the instructions as given. There are two ways to follow the directions:
   - Follow the instructions while remaining in ISPF Browse.
   - Record the data set name and member at the top of the screen and either print the instructions or use ISPF split screen and browse or edit the instructions while you follow them.
4. Fix any problems. If you encounter problems caused by your Customization Dialog values, modify your variables using the dialog, regenerate the instructions, and restart the customization process.

   Note: Remember that you cannot generate new customization jobs while either configuration data set is open!

You are done with this customization task when you have successfully followed the generated instructions.

Sample generated instructions: Federated application server node:

This article presents a sample of what the Customization Dialog's generated instructions might look like. This is a sample only--you must use the instructions generated from your own variables when configuring your system.

Note: This sample is based on the installation of WebSphere Application Server for z/OS Version 6.0.2. Generated instructions from different releases of Version 6 would have different content.

```
-----------------------------------------------
Instructions for customizing WebSphere for z/OS
for Federate stand-alone Application Server node.

The customization dialog has created jobs based on the information you
provided. These instructions tell you how to modify the operating
system and run the jobs to customize WebSphere for z/OS.

RULES:

1.  If you created the target data sets (*.CNTL and *.DATA) on another
    (driving) system, you must copy them to the target system and give
    them the same data set names.

2.  You must perform these instructions on your target system.

3.  Update TCP/IP by reserving the following ports for WebSphere for
    z/OS:

       SOAP JMX Connector port                    - 9360
       Node Discovery port                        - 7272
       Node Multicast Discovery Port              - 5000
       Node Agent's ORB port                      - 2809
       High Availability Manager Communication port - 9354
       Application Server's ORB Port              - 9810

    View member BBOTCPIA in

    'USERID.TODAY.CNTL'.

    Add the contents of this member to the PORT section of the file
    referenced by the DD statement for the TCP/IP profile in the
    TCP/IP start procedure. Cut and paste from this member into the
    data set used by your installation.

Note: The addNode process introduces a special utility server to
      the node.  This utility server is called a nodeagent and
      exists to support administrative functions on the node. By
      default the nodeagent takes over ORB port 2809.   Note on
      WebSphere z/OS the ORB port doubles as the INS CosNaming
      bootstrap port.  By default, this port (2809) was assigned
      to the Application Server.  Normally you want the nodeagent
      to be the INS CosNaming bootstrap point for the entire node,
      so that RMI/IIOP clients that do not override the INS
      CosNaming bootstrap defaults can locate within the namespace,
      EJBs installed on any server on that node.  In order for the
      nodeagent to take over port 2809,  the Application Server must
      be assigned a new ORB port.  The default new ORB port for the
      Application Server is 9810.  The nodeagent will take over an
      Application Server's ORB port if and only if the nodeagent's ORB
      port is equal to an Application Server's ORB port.  You can
      specify the nodeagent's ORB port in the 'ORB port' field.
      You can specify the new ORB port for the Application Server in
      the 'Appplication Server's ORB Port' field.

    ATTENTION: Skip this step if the ports are already defined in the
    TCP/IP profile.

4.  You must first complete the customization of a stand-alone
    application server and the customization of a deployment manager
    server before starting these instructions. Also, ensure that the
    deployment manager server has been started before starting these
    instructions.

+-----------+----------------------------------------------------------+
| --------  | If your system is busy, you may want to include a rule   |
+-----------+ in your WLM policy that OMVS work for job BBON001        |
| Done:     | (such as the postinstaller step) is to run in a service  |
|           | class with a high service objective.                     |
| By:       |                                                          |
+-----------+----------------------------------------------------------+
| BBOWADDN  | User ID requirement: WSADMIN                             |
+-----------+                                                          |
|           | If you choose the start the node agent automatically,    |
| Done:     | the user ID used to run BBOWADDN will also need the      |
|           | authority to issue the MVS START command.                |
|           |                                                          |
|           |                                                          |
|           | Add the application server(s) associated with the stand- |
|           | alone application server node to the deployment manager's|
|           | cell.                                                    |
| By:       |                                                          |
|           | ATTENTION: Before you run this job (BBOWADDN) to         |
|           | federate the stand-alone Application Server node, you    |
|           | must have started the stand-alone Application server     |
|           | that you are federating at least once so the applyPTF    |
|           | step runs.  Otherwise, the BBOWADDN job will fail with   |
|           | an error such as:                                        |
|           |   java.lang.NullPointerException at                      |
|           |   com.ibm.ws.management.tools.NodeFederationUtility      |
|           |     .createNDProductFile(NodeFederationUtility.java:1929)|
|           |                                                          |
|           | Note: Run this job only once for each node that you      |
|           | federate, regardless of how many application servers the |
|           | node contains.                                           |
|           |                                                          |
|           | RESULT: Upon successful completion of this job, you will |
|           | see the following message in SYSPRINT:                   |
|           |                                                          |
|           |   ADMU0003I: Node nodename has been successfully    |
|           |     federated.                                           |
|           |                                                          |
+-----------+----------------------------------------------------------+
| --------  | Start the node agent server                              |
+-----------+                                                          |
| Done:     | Note: The node agent is automatically started by the node|
|           | federation process.  This step is information, for       |
|           | starting the node agent at other times.                  |
| By:       |                                                          |
|           | If your Application Server proc name is XXXXXXXX         |
|           | your Application Server node name (short)                |
|           | is YYYYYYYY and your Deployment Manager cell             |
|           | name (short) is ZZZZZZZZ, issue the MVS command:         |
|           |                                                          |
|           |   START XXXXXXXX,JOBNAME=BBON001,                        |
|           |         ENV=ZZZZZZZZ.YYYYYYYY.BBON001                    |
|           |                                                          |
|           | This command starts the node agent server.               |
|           |                                                          |
|           | RESULT: The following message appears on the console and |
|           | in the job log of BBON001.                               |
|           |                                                          |
|           |   BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR    |
|           |   z/OS CONTROL PROCESS BBON001.                          |
|           |                                                          |
+-----------+----------------------------------------------------------+
| --------  | Start the Application Server (optional)                  |
+-----------+                                                          |
| Done:     | Note: You must start the node agent before you start the |
|           | Application Server, if the node agent has not already    |
|           | been started.                                            |
|           |                                                          |
| By:       | Example: If your Application Server proc name is         |
|           | XXXXXXXX, your Application Server node name (short) is   |
|           | YYYYYYYY your server short name is BBOS001, and your     |
|           | Deployment Manager cell name (short) is ZZZZZZZZ, issue  |
|           | the MVS command                                          |
|           |                                                          |
|           |   START XXXXXXXX,JOBNAME=BBOS001,                        |
|           |         ENV=ZZZZZZZZ.YYYYYYYY.BBOS001                    |
|           |                                                          |
|           | This command starts the Application Server.              |
|           |                                                          |
|           | RESULT: The following message appears on the console and |
|           | in the job log of BBOS001.                               |
|           |                                                          |
|           |   BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR    |
|           |   z/OS CONTROL PROCESS BBOS001                           |
|           |                                                          |
+-----------+----------------------------------------------------------+
```

Working with your new federated server node

Once you have successfully federated an application server node, perform the following tasks:

- Check the default host alias list and all other cell-level documents to see if any need to be added in support of the applications and application servers on the newly federated node. Cell-level documents are **NOT** automatically updated by the federation process.
- Remove the location service daemon port definitions for the stand-alone application server cell because these are not used after federation.

Note that Web server configurations in an unmanaged node in a stand-alone application server cell are not migrated as part of federation. Use the administrative console or scripting to add new Web Server definitions to a Network Deployment cell.

Once these tasks are accomplished, a federated application server node is just like any other application server node. The primary difference is that it already has an application server and applications if they were federated as well. See “Working with your new managed server node” on page 83 for further information.

Configuring with the Profile Management tool

Use this task to configure WebSphere Application Server for z/OS application serving environments for your z/OS target systems using the Profile Management tool.

- Choose a z/OS target system and complete the steps in Installing the product and additional software and Preparing the base operating system.
- Check that an FTP server is up and running on the z/OS target system.
- Choose a WebSphere Application Server for z/OS configuration (practice, stand-alone or Network Deployment cell) and complete the steps in Planning for product configuration.

Configuring a WebSphere Application Server for z/OS application serving environment consists of setting up the WebSphere Application Server for z/OS configuration directory for the environment, making any required changes to the z/OS target system that pertain to the particular application serving environment, and starting the new environment to verify the configuration. Configuring these application serving environments after product installation requires a fair amount of planning and coordination. If you have not previously configured WebSphere Application Server for z/OS, you should configure a ″practice″ stand-alone application server using the default options then proceed to configure the actual product configuration that you want. See “Using the Profile Management tool” and Building a practice WebSphere Application Server for z/OS cell for more information.

If you have already created a Network Deployment cell, follow the instructions in this section to expand the cell by creating a new managed node or federating an existing stand-alone application server node into the Network Deployment cell.

WebSphere Application Server for z/OS application serving environment nodes are created using the Profile Management Tool. Once a node is configured and running, make further changes using the Web-based administrative console or scripting.

After you have installed the WebSphere Application Server for z/OS product, prepared your z/OS target systems, and planned your new application server environment, perform the tasks in this section to configure and start the application server environment.

1. Install the Application Server Toolkit (AST) by following the instructions in “Installing the Application Server Toolkit” on page 95.
2. Review the procedures for the tool in “Using the Profile Management tool.”
3. Follow the directions for the type of application server environment you want to configure:
   - “Creating a stand-alone application server cell” on page 97
   - “Creating a network deployment cell” on page 97
   - “Creating a Network Deployment cell with an application server” on page 98
   - “Creating a managed node” on page 111
   - “Federating a stand-alone application server into a network deployment cell” on page 112

Once your application serving environment is up and running, you can install and test the applications. You might also want to configure your Web servers to interact with WebSphere Application Server for z/OS.

Using the Profile Management tool

This article provides general information on starting and using the Profile Management tool. See the instructions for each customization task for directions on using the Profile Management tool to perform a particular task.

The WebSphere Application Server for z/OS Profile Management tool (PMT) is a dialog tool that runs on the WebSphere Application Server Toolkit.

The WebSphere Application Server for z/OS Profile Management Tool (PMT) is an Eclipse plug-in, running under the Application Server Toolkit, that you use for the initial setup of WebSphere Application Server for z/OS cells and nodes. The Profile Management Tool itself does not create the cells and nodes; instead, it creates batch jobs, scripts, and data files that you can use to perform WebSphere Application Server for z/OS customization tasks. These jobs, scripts, and data files form a **customization definition** on your workstation, which is then uploaded to z/OS and used for customization.

Note: In WebSphere Application Server for z/OS, you use the ISPF-based Customization Dialog or the Eclipse-based Profile Management Tool (and the jobs they generate) to create new cells and nodes. Once you have created a stand-alone application server or Network Deployment cell, however, you use the WebSphere Application Server for z/OS administrative console or scripting to administer it.

The Profile Management tool is intended for use by a systems programmer or WebSphere Application Server for z/OS administrator who is familiar with the z/OS target system on which the resulting WebSphere Application Server for z/OS cells and nodes will run.

The PMT uses response files to hold the various values used to create WebSphere Application Server for z/OS customization jobs, scripts and files. These response files remain on the workstation where the Profile Management tool is run.

Follow these steps to use the Profile Management tool:

1. To install the Application Server Toolkit (AST), which includes the Profile Management tool, refer to "Installing the Application Server Toolkit" for more information.
2. To start the Profile Management tool, see the instructions in "Starting the Profile Management tool" in order to use the tool.
3. To create a customization definition, follow the steps in "Creating the customization definition."
4. To review a customization definition, follow the steps in "Reviewing the customization definition" on page 96.
5. To upload customization jobs to the target z/OS system, follow the instructions in "Uploading the customization jobs to z/OS" on page 96.
6. To delete a customization definition, select the customization definition and click **Delete**.

Installing the Application Server Toolkit

This article leads you through the tasks involved in installing the WebSphere Application Server Toolkit.

The Application Server Toolkit is available on CD-ROM in the WebSphere Application Server CD-ROM package.

Follow the steps below to install the WebSphere Application Server toolkit.

1. Install the Application Server Toolkit. Before you can work with the Profile Management tool, install the Application Server Toolkit. The Application Server Toolkit is available on CD-ROM in the WebSphere Application Server CD-ROM package. To install Application Server Toolkit, follow the installation instructions that are on its CD-ROM. Install the Application Server Toolkit on the Windows® platform.

   For more information refer to the section ″Installing the Application Server Toolkit″ in the *Installing your application serving environment* PDF.
2. Start working with the Profile Management tool. After the Application Server Toolkit is installed, you are ready to start working with the Profile Management tool.

Starting the Profile Management tool

This article leads you through the tasks involved in starting the Application Server Toolkit.

Install the Application Server Toolkit (AST) on your workstation. Refer to *Installing the Application Server Toolkit* for instructions.

Follow the steps below:

1. Launch the Application Server Toolkit: On Windows®: From the **Start** button, select **Programs** or **All Programs**. Select **IBM WebSphere** and then select **Application Server Toolkit Version 6.1**. Finally, select **Application Server Toolkit**.
2. Select a workspace, unless you have previously selected a default workspace.
3. To start the Profile Management tool, click on **Window** and release the mouse button on **Preferences**. From the navigation bar on the left, find **WebSphere for z/OS**, located under the Server section. Select **WebSphere for z/OS**
4. The table titled ″WebSphere for z/OS customization definition″ shows the customization definitions in the current workspace. You are now ready to select and work with a customization definition.

Creating the customization definition

This article leads you through the tasks involved in creating the customization profile.

Install the Application Server toolkit and perform any planning indicated for the customization task you have chosen.

A customization definition consists of a set of files on your workstation that is uploaded to z/OS and used to perform customization tasks.

1. Start the Profile Management tool. Refer to “Starting the Profile Management tool” on page 95 to start the tool.
2. From the WebSphere for z/OS Customization Profile menu, select the **create** button. Read the Welcome page and select **Next**.
3. Select the type of customization definition you wish to create, and click **Next**.
4. The Profile Management tool will generate a name for this customization definition, and choose a directory to store the customization definition files. Change these values if desired. Also, if you need to use an existing response file to pre-load customization values, enter the path to the response file where indicated. When you have finished, click **Next**.
5. Enter the high level qualifiers for the pair of z/OS data sets that will contain the customization jobs and files for this customization task, when the definition is uploaded to z/OS. When you have finished, click **Next**.
6. Proceed through the panels as displayed, using the **Back** and **Next** buttons. Change the customization values where necessary. Each field has help information that can be displayed by hovering the cursor over the field. Use the **Cancel** button to leave the profile creation process without creating a definition.
7. When you have successfully filled in all the customization panels for this customization profile type, the Profile Management tool will display the definition type, location, and name. Click **Create** to build the customization definition jobs and files on your workstation.
8. Click **Finish** to return to the WebSphere for z/OS Customization Definition menu.

Reviewing the customization definition

This article explains how to work with a customization definition that you have created in the Profile Management tool.

In order to review the customization definition, you will need to have the Application Server Toolkit installed.

Follow these steps to review profiles that you have created with the Profile Management tool.

1. Refer to “Starting the Profile Management tool” on page 95 for instructions.
2. From the WebSphere for z/OS Customization Definition menu, select one of the customization profiles you have created.
3. If you want to view the selected profile, click **View**. You will be presented with a Customization Profile Information window with three tabs:
   a. The **Summary** tab displays the definition name and type, the location on your workstation of various files related to the customization definition, and the z/OS data set names that will be used for the customization jobs and files when they are uploaded.
   b. The **Instructions** tab displays the generated task instructions you will follow after the jobs and files are uploaded to z/OS.
   c. The **Response File** tab displays the customization values stored in the response file.
4. If you want to update or change the selected profile, click **Regen**.
5. If you need to delete the selected profile, click **Delete**.
6. When you have finished reviewing the profile, click **Finish** or **Cancel** to return to the *WebSphere for z/OS customization definition* panel.

Uploading the customization jobs to z/OS

This article explains how to upload jobs and files associated with a customization definition to z/OS.

In order to upload the customization definition, you will need to have the Application Server Toolkit installed on your workstation.

Follow the steps below to upload to z/OS.

1. Refer to “Starting the Profile Management tool” on page 95 for details in starting the tool.
2. From the WebSphere for z/OS Customization Definition menu, select the customized definition you want uploaded to z/OS by highlighting the line.
3. Press the **Upload** button to activate the step.
4. On the next dialog menu, you will be prompted to select the target z/OS system, along with your userid and password. The tool will list the designated partitioned datasets where the profiles will be stored. If you are uploading to a system different from the system where the jobs will be run, then you need to target a volume which is shared with that system.
5. Indicate whether the target datasets need to be allocated on z/OS, optionally specify the volume and unit type.
6. Press the **Upload** button to upload the files to z/OS. Wait until the upload step has completed. Once the upload has completed, you will need to press the **OK** button on the popup window.
7. Press the **Finish** button to end the Profile Management tool.

Creating a stand-alone application server cell

This article leads you through the tasks involved in setting up a WebSphere Application Server for z/OS stand-alone application server environment using the Profile Management tool.

Have available copies of the worksheets that you completed as a part of Planning for a stand-alone application server cell and Planning for product configuration.

Follow the steps below to set up a new WebSphere Application Server for z/OS stand-alone application server cell.

1. Create a customization profile for the stand-alone application server.
   a. Follow the instructions in “Creating the customization definition” on page 95.
   b. Select ″z/OS Application Server″ for application environment type.
   c. Fill in the prompts using the values from your Customization Dialog worksheet: Stand-alone application server cell.
2. Review the stand-alone application server definition to make sure all the values are correct. Refer to the “Reviewing the customization definition” on page 96 for instructions.
3. Upload the customization jobs and scripts to the target z/OS system. Refer to “Uploading the customization jobs to z/OS” on page 96 for instructions.
4. Follow the instructions in the BBOSSINS member of the CNTL data set you uploaded. See “Following the generated customization instructions: Stand-alone application server cell” on page 33 for more details.

When you have successfully completed the steps in the generated instructions. The new stand-alone application server is up and running on the chosen z/OS system. See “Working with your new server” on page 44 for more information.

You can now deploy and test applications on your new stand-alone application server.

Creating a network deployment cell

This article leads you through the tasks involved in setting up a WebSphere Application Server for z/OS network deployment cell using the Profile Management tool.

Have available copies of the worksheets that you completed as a part of Planning for a Network Deployment cell and Planning for product configuration.

Perform this task to set up a new WebSphere Application Server for z/OS Network Deployment cell.

1. Create a customization definition for the network deployment cell.
   a. Follow the instructions in “Creating the customization definition” on page 95.
   b. Select ″z/OS deployment manager″ for application environment type.
   c. Fill in the prompts using the values from your Customization Dialog worksheet: Network Deployment cell.
2. Review the network deployment definition to make sure all the values are correct. Refer to the “Reviewing the customization definition” on page 96 for instructions.
3. Upload the customization jobs and scripts to the target z/OS system. Refer to “Uploading the customization jobs to z/OS” on page 96 for instructions.
4. Follow the instructions in the BBOCCINS member of the CNTL data set you uploaded. Refer to “Following the generated customization instructions: Network Deployment cell” on page 54 for more details.

When you have successfully completed the steps in the generated instructions the task is finished. The new network deployment cell is up and running on the chosen z/OS system. See “Working with your new deployment manager” on page 64 for more information.

Add application server nodes to your cell using one of two methods:

- Create a new managed node using the Profile Management tool and add application servers to it using the administrative console or scripting.
- Federate existing stand-alone application servers into your network deployment cell to create managed nodes with application servers.

Creating a Network Deployment cell with an application server

This article leads you through the tasks involved in setting up a WebSphere Application Server for z/OS Network Deployment cell including an initial application server, using the Profile Management tool.

Have available copies of the worksheets that you completed as a part of Planning for a Network Deployment cell and Planning for product configuration.

Perform this task to set up a new WebSphere Application Server for z/OS Network Deployment cell (deployment manager and federated application server). These steps will lead you through the process.

1. Create a customization definition for the network deployment manager.
   a. Follow the instructions in “Creating the customization definition” on page 95.
   b. Select ″z/OS cell (deployment manager and an application server)″ for application environment type.
   c. Fill in the prompts using the values from your “Customization Dialog worksheet: Network Deployment cell with an Application Server” on page 106 and “Customization Dialog variables: Network Deployment cell with an Application Server” on page 99.
2. Review the Network Deployment cell definition to make sure all the values are correct. Refer to the “Reviewing the customization definition” on page 96 for instructions.
3. Upload the customization jobs and scripts to the target z/OS system. Refer to “Uploading the customization jobs to z/OS” on page 96 for instructions.
4. Follow the instructions in the BBODMINS member of the CNTL data set you uploaded. See “Following the generated customization instructions: Network Deployment cell” on page 54 for more details.

When you have successfully completed the steps in the generated instructions the task is complete. The new Network Deployment cell is up and running on the chosen z/OS system. See “Working with your new deployment manager” on page 64 for more information.

Add additional application server nodes to your cell using one of two methods:

- Create a new managed node using the Profile Management tool and add application servers to it using the administrative console or scripting.
- Federate existing stand-alone application servers into your network deployment cell to create managed nodes with application servers.

Customization Dialog variables: Network Deployment cell with an Application Server

This article lists definitions for the terms you will come across in the WebSphere Application Server for z/OS Customization Dialog and the Profile Management tool..

The WebSphere Application Server for z/OS runtime requires four stand-alone cell servers: application server, deployment manager, node agent, and location service daemon. The panels corresponding to the following tables set up the names, network configuration, start procedures, and user IDs for a Network Deployment cell with an application server.

High Level Qualifier

Configure Common Groups and Users

System Locations

This section identifies the target z/OS system on which you will configure the deployment manager for the Network Deployment cell, along with system data set names.

System name
The system name for the target z/OS system on which you will configure WebSphere Application Server for z/OS.

Sysplex name
The sysplex name for the target z/OS system on which you will configure WebSphere Application Server for z/OS.

Tip: If you are not sure what the system name (&SYSNAME) and sysplex name (&SYSPLEX) are, use the console command `D SYMBOLS` on the target z/OS system to display them.

For the following, specify the fully qualified data set names without quotes.

Rule: You can specify up to 44 characters for the data set names.

PROCLIB
An existing procedure library where the WebSphere Application Server for z/OS cataloged procedures are added.

WebSphere product data sets high level qualifier

WebSphere Application Server for z/OS product data sets

Specify the following WebSphere Application Server for z/OS libraries so they can be accessed by the customized jobstreams the dialog produces. These data sets must be cataloged. See the section ″Product data sets″ in the *Installing your application serving environment* PDF for more information.

Run WebSphere Application Server from STEPLIB (Y/N)?
Specifies whether to load WebSphere Application Server for z/OS load modules from STEPLIB (″Y″) or from the link pack area and link list (″N″).

See the section ″Link pack area, link list, and STEPLIB″ in the *Installing your application serving environment* PDF for more information.

Note: Specify ″Y″ if you have another instance of WebSphere Application Server for z/OS (Version 4 or later) in the system link pack area or link list.

SBBOLPA
WebSphere Application Server for z/OS load module library. It has modules that should go into LPA or the location service daemon STEPLIB.

SBBOLOAD
WebSphere Application Server for z/OS 31-bit load module library. It has members that should go into the link list or LPA, or into STEPLIB.

SBBGLOAD
WebSphere Application Server for z/OS 64-bit load module library. It has members that should go into the link list or LPA, or into STEPLIB.

SBBOLD2
WebSphere Application Server for z/OS load module library that you installed through SMP/E. It has members that should go into the link list, or into STEPLIB. **DO NOT** place them in LPA.

SBBOEXEC
WebSphere Application Server for z/OS CLIST library.

SBBOMSG
SBBOMSG WebSphere Application Server for z/OS message skeletons for language translation.

Locations of File System Resident components

WebSphere Application Server product directory
The name of the directory where WebSphere Application Server for z/OS files reside after installation.

See the section ″Product file system″ in the *Installing your application serving environment* PDF for more information.

WebSphere Configuration File System

This section defines the WebSphere configuration HFS that you will use for the deployment manager.See the section ″Configuration file system″ in the *Installing your application serving environment* PDF for more information.

Mount point
Read/write HFS directory mount point where application data and environment files are written. The customization process creates this mount point if it does not already exist.

Name Hierarchical File System data set you will create and mount at the above mount point.

Rule: You can specify up to 44 characters for the data set name.

Volume, or '*' for SMS
Specify either the DASD volume serial number to contain the above data set or ″*″ to let SMS select a volume. Using ″*″ requires that SMS automatic class selection (ACS) routines be in place to select the volume. If you do not have SMS set up to handle data set allocation automatically, list the volume explicitly.

Primary allocation in cylinders
Initial size allocation in cylinders for the above data set.

Recommendation: The minimum suggested size is 420 cylinders (3390).

Secondary allocation in cylinders
Size of each secondary extent in cylinders.

Recommendation: The minimum suggested size is 100 cylinders.

Use zSeries® file system or HFS

WebSphere error log stream

This section defines the System Logger log stream that you will use for WebSphere Application Server for z/OS error logging if TRACEBUFFLOC=BUFFER. Having the error log in the log stream is optional. By default, the error log is written to JES SYSOUT.

Name Name of the WebSphere error log stream you will create.

Rules:
- The name must be 26 or fewer characters.
- Do NOT put quotes around the name.

Ctrace Writer Definitions

WebSphere application Server for z/OS uses component trace (CTRACE) to capture and to display trace data in trace data sets. WebSphere Application Server for z/OS identifies itself to CTRACE with the *cell short name.*

Trace Parmlib member suffix
Value that is appended to CTIBBO to form the member name for the Trace parmlib member.

Server customization

During this customization task, you will create a cell configuration, a deployment manager node and server, and a location service daemon. The panels corresponding to the following tables set up the names, network configuration, start procedures, and user IDs for a deployment manager server.

Rule: In the following customization, names must be eight or fewer characters unless otherwise specified.

Cell name (short)
Name that identifies the cell to z/OS facilities such as SAF.

Rules:
- Name must be eight or fewer characters and all uppercase.
- Name must be unique among all other cells in the sysplex.

Cell name (long)
Primary external identification of this WebSphere Application Server for z/OS cell. This name identifies the cell as displayed through the administrative console.

Rules:
- Name must be 50 or fewer characters and can be of mixed case.
- Name must be unique among all other cells in the sysplex.

Deployment manager node name (short)
Name that identifies the deployment manager node to z/OS facilities such as SAF.

Rules:
- Name must be eight or fewer characters and all uppercase.
- Name must be unique within the cell.

Deployment manager node name (long)
Primary external identification of this WebSphere Application Server for z/OS deployment manager node. This name identifies the node as displayed through the administrative console.

Rules:
- Name must be 50 or fewer characters and can be of mixed case.
- Name must be unique within the cell.

Deployment Manager Server name (short)
This value identifies the deployment manager server to z/OS facilities such as SAF.

Note: The server short name is used as the server JOBNAME.

Rule: Name must usually contain seven or fewer all-uppercase characters. To change the jobname later to an eight-character value, you must follow the steps outlined in “Converting a 7-character server short name to 8 characters” on page 284.

Deployment Manager Server name (long)
Name of the application server and the primary external identification of this WebSphere

Application Server for z/OS server. This name identifies the server as displayed through the administrative console. The server name has a fixed name (long) of ″dmgr″.

Deployment Manager Cluster transition name

WLM APPLENV name for this server.

Note: The deployment manager is not clusterable, so this value never actually becomes the cluster short name of this server's cluster. However, like an application server, the deployment manager still needs an APPLENV, so the cluster transition name is used for this purpose.

Rule: Name must be eight or fewer characters and all uppercase.

Application Server Cluster Transition name

Node Agent and Application Server node names (short)

Node Agent and Application Server node names (long)

Node Agent Server name (short)

Node Agent Server name (long)

Application Server name (short)

Application Server name (long)

Deployment manager definitions

Rule: In the following definitions, names must be eight or fewer characters unless otherwise specified.

Controller Information

Jobname

The jobname, specified in the MVS START command JOBNAME parameter, associated with the deployment manager controller. This is the same as the server short name and it cannot be changed through the Customization Dialog.

Procedure name

Name of member in your procedure library to start the deployment manager controller.

Rule: Name must be seven or fewer characters.

User ID

The user ID associated with the deployment manager controller.

Note: If you are using a non-IBM security system, the user ID might have to match the procedure name. Please refer to your security system's documentation.

UID The user identifier associated with this user ID.

Rule: UIDs must be unique numbers, between 1 and 2,147,483,647, within the system.

Servant Information

Jobname

The jobname used by WLM to start the deployment manager servant. This is set to the server short name followed by the letter ″S″, and it cannot be changed through the Customization Dialog.

Procedure name

Name of member in your procedure library to start the deployment manager servant.

Rule: Name must be seven or fewer characters.

User ID

The user ID associated with the deployment manager servant.

Note: If you are using a non-IBM security system, the user ID might have to match the procedure name. Please refer to your security system's documentation.

UID The user identifier associated with this user ID.

Rule: UIDs must be unique numbers, between 1 and 2,147,483,647, within the system.

Deployment manager TCP/IP information

Note: Do not choose port values already in use.

Node host name
IP name or address of the system on which the server is configured. This value is used by other WebSphere Application Server for z/OS functions to connect to this server.

Note: The node host name must always resolve to an IP stack on the system where the application server runs. (You can either have multiple IP stacks on a given MVS image and have the deployment manager and stand-alone application server tied to separate host names, or you can associate them with the same node host name.) The node host name cannot be a DVIPA or a DNS name that, in any other way, causes the direction of requests to more than one system.

SOAP JMX Connector port
Port number for the JMX HTTP connection to this server based on the SOAP protocol. JMX is used for remote administrative functions, such as invoking scripts through wsadmin.sh.

Rule: Value cannot be 0.

Cell Discovery Address port
Port number used by node agents to connect to this deployment manager server.

ORB Listener host name
IP address on which the server's ORB listens for incoming IIOP requests. The default is ″*″, which instructs the ORB to listen on all available IP addresses.

ORB port
Port for IIOP requests that acts as the bootstrap port for this server and also as the port through which the ORB accepts IIOP requests.

Rule: Value cannot be 0.

ORB SSL port
Port for secure IIOP requests. The default is ″0″, which allows the system to choose this port.

HTTP transport host name
IP address on which the server's Web container should listen for incoming HTTP requests. The default is ″*″, which instructs the Web container to listen on all available IP addresses.

Note: The ″transport host name″ becomes the ″hostname″ in the virtualhosts.xml file, which makes setting a specific IP address here less than ideal because, if you do so, you are restricting yourself to that IP address until you go into the administrative console and add another virtual host.

Administrative console port
Port for HTTP requests to the administrative console.

Administrative console secure port
Port for secure HTTP requests to the administrative console.

High Availability Manager communication port
Port on which the High Availability Manager listens.

Location service daemon

The location service daemon is the initial point of client contact in WebSphere Application Server for z/OS. The server contains the CORBA-based location service agent, which places sessions in a cell. All RMI/IIOP IORs (for example, for enterprise beans) establish connections to the location service daemon first, then forward them to the target application server.

Daemon home directory
Directory in which the location service daemon resides. This is set to the configuration HFS mount point/Daemon and cannot be changed.

Daemon jobname
Specifies the jobname of the location service daemon, specified in the JOBNAME parameter of the MVS start command used to start the location service daemon.

Note: The same daemon jobname is used on all z/OS systems hosting the cell. Therefore, this name is generic and has no system indicator. When the deployment manager's controller started, its location service daemon was started automatically using this particular daemon jobname value.

Caution: When configuring a second cell, ensure you change the daemon jobname from the default or value you used for the first cell.

Note: A server automatically starts the location service daemon if it is not already running.

Procedure name
Name of the member in your procedure library to start the location service daemon.

Rule: Name must be seven or fewer characters.

User ID
The user ID associated with the location service daemon.

UID The user identifier associated with this user ID.

Rule: UIDs must be unique numbers, between 1 and 2,147,483,647, within the system.

IP Name
The fully qualified IP name, registered with the Domain Name Server (DNS), that the location service daemon uses.

Adhere to the following regarding your location service daemon IP name:

- It must be unique in the sysplex.
- It must not have the same value as any one system's node host name. (You can use the host name of the LPAR.)
- It should be a name that can be used in conjunction with a routing service that distributes requests among nodes in the cell (that is, systems in the sysplex).
- It should be a virtual IP address (VIPA) if you are operating in a sysplex.

Note:

- IBM recommends you use z/OS Sysplex Distributor by way of a ″Dynamic Virtual IP address″ (DVIPA). See related sections in the information center for more information.
- Select the IP name for the location service daemon carefully. Once chosen it is difficult to change, even in the middle of customization. This name must not be a numeric, such as, 3.7.25.43.

Daemon Listen IP
The default value is *.

Rule: The default is * or a numeric IP address.

Port The port number on which the location service daemon listens.

Note: Select the port number for the location service daemon carefully. You can choose any port you want; but, once chosen, it is difficult to change, even in the middle of customization.

SSL port
The port number on which the location service daemon listens for SSL connections.

Register daemon with WLM DNS
If you use the WLM DNS (connection optimization), you must select ″Y″ to register your location service daemon with it. Otherwise, select ″N.″

Note: Only one location service daemon per LPAR can register its domain name with WLM DNS. If you have multiple cells in the same LPAR and register one location service daemon and then a second, the second will fail to start.

SSL customization

If you plan to enable Administrative Security at some point, as is recommended, fill in the following SSL values:

Certificate authority keylabel
The name of the key label that identifies the certificate authority (CA) to be used in generating server certificates.

Generate authority certificate
Select ″Y″ to generate a new CA certificate. Select ″N″ to have an existing CA certificate generate server certificates.

Expiration date for certificates
The expiration date used for any X509 Certificate Authority certificates, as well as the expiration date for the personal certificates generated for WebSphere Application Server for z/OS servers. You must specify this even if you selected ″N″ for ″Generate Certificate Authority (CA) certificate.″

Default RACF® keyring name
The default name given to the RACF keyring used by WebSphere Application Server for z/OS. The keyring names created for repertoires are all the same within a cell.

Enable SSL on location service daemon
Select ″Y″ if you want to support secure communications using Inter-ORB Request Protocol (IIOP) to the location service daemon using SSL. If you specify ″Y″, a RACF keyring will be generated for the location service daemon to use.

Security Customization

During the initial setup, you can choose one of the following three options for administrative security.

Option 1 - WebSphere for z/OS security
Use the z/OS system’s SAF-compliant security database to define WebSphere Application Server users. The EJBROLE profile will be used to control role-based access to applications. An administrator user ID and an unauthenticated user ID will be created and defined in the security database. Select this option if the WebSphere Application Server environment will run entirely on z/OS with a shared SAF-compliant (Local OS) user registry, or if you plan to implement an non-Local OS user registry (such as LDAP) with mapping to SAF user IDs.

Option 2 - WebSphere family security
Use a simple file-based registry to define WebSphere Application Server users. An administrator user ID will be created and defined in the file-based registry.

Option 3 - No security
Do not enable administrative security (not recommended). Your WebSphere Application Server environment will not be secured until you configure and enable security manually.

Depending on the security option you choose, there may be additional customization dialog values you need to set.

Security customization - WebSphere for z/OS security

For this security option, you must decide whether to set a security domain name, and choose an administrator user ID and an unauthenticated (guest) user ID.

Use security domain identifier in RACF profiles
Set this to Y if you wish to include a security domain name in certain SAF security checks (APPL, CBIND, EJBROLE). Enter a 1-8 character security domain name.

Administrator user ID
Administrator user ID Enter a valid SAF user ID which will become the initial cell administrator. If this user ID already exists, it must have the WebSphere Application Server configuration group for this cell as its default Unix System Services group. Also specify a valid UID for this user ID.

Unauthenticated User ID
Enter a valid SAF user ID which will associated with unauthenticated client requests. Also specify a valid UID for this user ID.

Security customization - WebSphere family security

For this security option, you must choose an administrator user ID and password.

Administrator user ID
Enter an alphanumeric user ID which you will use to log on to the administrative console and perform administrative tasks. This user ID and its password will initially be the only entry in the file-based user registry.

Administrator password
This password must not be blank.

Security Customization - no security

For this security option, there are no other choices to make. Your WebSphere Application Server environment will not be secured until you configure and enable security manually.

Web Server Configuration

If you want to create a web server definition at this point, fill in the following values.

Note: You can only have one Web server defined on a stand alone application server.

You will not be able to administer the Web server through the integrated solutions console on the stand alone server until it is federated to a managed node.

Web Server Type
Valid values: IHS, HTTPSERVER_ZOS, APACHE, IPLANET, DOMINO, IIS.

Web Server Operating System
The operating system where the Web server is located.

Web Server Name
The name used in defining the Web server in the admin console.

Web Server Host or IP Address
IP name or address of the z/OS on which the Web server is located.

Web Server Port
HTTP Port on which the Web server is listening.

Web Server Install Directory Path
/usr/lpp/internet/

Web Server Plugin Install Directory Path
(*This varies by Web server type*)

Customization Dialog worksheet: Network Deployment cell with an Application Server

Date: ______________________

Purpose of this Network Deployment cell:
____

System name: _________________________ Sysplex name: _________________________

Print out this worksheet and use it when collecting information about the WebSphere Application Server for z/OS Customization Dialog variables. The variables and defaults are provided, along with spaces for you to fill in your own value for each.

z/OS Target Dataset High Level Qualifier

Configure Common Groups and Users

System Locations

Item	Default	Your value
System Name		
Sysplex Name		
PROCLIB	SYS1.PROCLIB	
WebSphere product data set high level qualifier		

WebSphere Application Server for z/OS product data sets

Run WebSphere Application Server from STEPLIB (Y/N)?	Y	
SBBOLPA	*was_hlq*.SBBOLPA	
SBBOLOAD	*was_hlq*.SBBOLOAD	
SBBGLOAD	*was_hlq*.SBBGLOAD	
SBBOLD2	*was_hlq*.SBBOLD2	
SBBOEXEC	*was_hlq*.SBBOEXEC	
SBBOMSG	*was_hlq*.SBBOMSG	

Locations of Resident File System Components

WebSphere Application Server product directory	/usr/lpp/zWebSphere/V6R1	

WebSphere Configuration File System

Mount point	/WebSphere/V6R1	
Name	OMVS.WAS.CONFIG.HFS	
Volume, or '*' for SMS	*	
Primary allocation in cylinders	420	
Secondary allocation in cylinders	100	
Use zSeries File Ssystem instead of HFS	Y	

WebSphere error log stream

Name	WAS.ERROR.LOG	

Component Trace

Trace Parmlib member suffix	60	

Server customization

Cell name (short)	*sysplexname*	
Cell name (long)	*sysplexname*Network	
Deployment manager node name (short)	*sysplexname*	
Deployment manager node name (long)	*sysplexname*Manager	
Deployment manager server name (short)	BBODMGR	
Deployment manager server name (long)	dmgr	(cannot change)
Node agent & application server node name (short)	*systemname*	
Node agent & application serve node name (long)	*systemname*	
Node agent server name (short)	BBON001	
Node agent server name (long)	nodeagent	
Application server name (short)	BBOS001	
Application server name (long)	server1	
Deployment manager cluster transition name	BBODMGR	
Application server cluster transition name	BBOC001	

Server customization

Deployment manager path related to mount point	DeploymentManager	
Application server path related to mount point	AppServer	
Admin asynch operations proc name	BBOW6SH	
Asynch ID	WSADMSH	
Asych UID	2504	
Install samples	Y	

Server address space information

Application Server Controller Information		
Jobname	*servershortname*	(cannot change)
Procedure name	BBO6ACR	
User ID	ASCR1	

UID	2431	
Application Server Control Region Adjunct		
Jobname	*servershortname*A	(cannot change)
Procedure name	BBO6CRA	
User ID	ASCRA1	
UID	2433	
Application Server Servant Information		
Jobname	*servershortname*S	(cannot change)
Procedure name	BBO6ASR	
User ID	ASSR1	
UID	2432	

Deployment manager definitions

Deployment Manager Controller Information		
Jobname	*deploymentmanagershortname*	(cannot change)
Procedure name	BBO6DCR	
User ID	DMCR1	
UID	2421	
Deployment Manager Servant Information		
Jobname	*deploymentmanagershortname*S	(cannot change)
Procedure name	BBO6DSR	
User ID	DMSR1	
UID	2422	

Deployment manager TCP/IP information

Node host name	(blank)	
SOAP JMX connector port	8879	
Cell Discovery Address port	7277	
ORB Listener host name	*	
ORB port	9809	
ORB SSL port	0	
HTTP transport host name	*	
Administrative console port	9060	
Administrative secure console port	9043	
High availability manager communication port	9352	

Node Agent TCP/IP information

JMX SOAP Connector port	9360	
ORB port	2810	

ORB SSL port	0	
High Availability Manager Communication Port	9354	
Node Discovery Port	7272	
Node Multicast Discovery Port	5000	

Application Server TCP/IP information

JMX SOAP Connector port	8880	
ORB port	2809	
ORB SSL port	0	
HTTP port	9080	
HTTP SSL port	9443	
High Availability Manager Communication Port	9353	
Service Integration port	7276	
Service Integration Secure port	7286	
Service Integration MQ Interoperability port	5558	
Service Integration MQ Interoperability Secure port	5578	
Session Initiation Protocol	5060	
Session initiation Secure Protocol	5061	

Location service daemon

Daemon home directory	*configmountpoint*/Daemon	(cannot change)
Daemon jobname	BBODMNC	
Procedure name	BBO6DMN	
User ID	WSDMNCR1	
UID	2411	
IP name	*nodehostname*	
Port	5755	
SSL port	5756	
Register daemon with WLM DNS	N	

SSL Customization

Certificate Authority Keylabel	WebSphereCA
Generate Certificate Authority (CA) certificate	Y
Expiration date for CA authority	2010/12/31
Default RACF keyring name	WASkeyring.*node name*
Enable SSL on location service daemon	Y

Security configuration

WebSphere for z/OS security

Use security domain identifier in RACF profiles:	N
Security domain identifier.	
WebSphere Application Server Administrator Information	
User ID	WSADMIN
UID	2403
WebSphere Application Server Unauthenticated User Information	
User ID	WSGUEST
UID	2402

WebSphere family security

User ID.: WSADMIN	WSADMIN
Password:	
Confirm Password:	Insert link to security section.

Web Server Configuration

Web Server Type	HTTPSERVER - Z/OS	
Web Server Operating System	os390	
Web Server Name	webserver1	
Web Server Host or IP Address	*nodehostname*	
Web Server Port	80	
Web Server Install Directory Path	(varies by type)	
Web Server Plugin Install Directory Path	(varies by type)	

Job statement definition

(ACCTNO,ROOM),'USERID',CLASS=A,REGION=OM
//*
//*
//*

Name of Network Deployment cell configuration variable data set:
____

Creating a managed node

This article leads you through the tasks involved in creating a customization definition for a managed server node using the Profile Management tool.

Perform this task to create a customization definition for the managed node.

1. Create a customization definition for the managed node.
   a. Follow the instructions in “Creating the customization definition” on page 95,
   b. Select ″z/OS managed (custom) node″ for application environment type.
   c. Fill in the prompts using the values from your Customization Dialog worksheet: Managed node.
2. Review the managed node definition to make sure all the values are correct. Refer to the “Reviewing the customization definition” on page 96 for instructions.
3. Upload the customization jobs and scripts to the target z/OS system. Refer to “Uploading the customization jobs to z/OS” on page 96 for instructions.
4. Follow the instructions in the BBOMNINS member of the CNTL data set you uploaded. See “Following the generated customization instructions: Managed node” on page 73 for more details.

When you have successfully completed the steps in the generated instructions this task is complete. The new managed node is up and running on the chosen z/OS system. See “Working with your new managed server node” on page 83 for more information.

Federating a stand-alone application server into a network deployment cell

This article leads you through the tasks involved in federating a WebSphere Application Server for z/OS stand-alone application server into a network deployment cell using the Profile Management tool.

Perform this task to federate an existing WebSphere Application Server for z/OS stand-alone application server into a Network Deployment cell.

1. Create a customization definition for federating the stand-alone application server into a network deployment cell.
   a. Follow the instructions in “Creating the customization definition” on page 95.
   b. Select ″z/OS federate an Application Server″ for application environment type.
   c. Fill in the prompts using the values from your Customization Dialog worksheet: Federating an application server node.
2. Review the definition to make sure all the values are correct. Refer to the “Reviewing the customization definition” on page 96 for instructions.
3. Upload the customization jobs and scripts to the target z/OS system. Refer to “Uploading the customization jobs to z/OS” on page 96 for instructions.
4. Follow the instructions in the BBOANINS member of the CNTL data set you uploaded. See “Following the generated customization instructions: Federated application server node” on page 90 for more details.

When you have successfully completed the steps in the generated instructions this task is complete. The new federated application server node is up and running on the chosen z/OS system. See “Working with your new federated server node” on page 93 for more information.

Using the installation verification test

You initially run the installation verification test (IVT), which verifies that WebSphere Application Server is configured correctly for your system, during ISPF customization of each of your systems. If you want to run the IVT at a time other than during initial customization, however, there are two methods from which you can choose.

Note: These options are now available when you are running a stand-alone application server configuration as well as after federating an application server.

Select either method to invoke the IVT:

- "Running the installation verification test with a job" on page 45
- "Running the installation verification test from a command line" on page 45

Running the installation verification test with a job

The application server must be running.

Follow these steps to run the installation verification test (IVT) using the BBOWIVT job.

1. Verify that the application server is running.
2. Confirm that the ivtApp application is installed and started.
3. Submit the job BBOWIVT.

The IVT runs a series of verification tests and reports pass or fail status for each in the messages generated by the BBOWIVT job. The output is written to standard output and to the `WAS_HOME/profiles/default/logs/ivtClient.log` file.

Running the installation verification test from a command line

The application server must be running.

Follow these steps to run the installation verification test (IVT) from a command line.

1. Verify that the application server is running.
2. Confirm that the ivtApp application is installed and started.
3. From a command line, navigate to the `WAS_HOME/bin` directory.
4. Issue the following command:

   ```
   ivt.sh server_name profile_name -p port_number [-host host_name]
   ```

   where

   - *server_name* is the short name of the server.
   - *profile_name* is the name of the profile.
   - `-p` *port_number* is an argument that specifies the port number.
   - `-host` *host_name* is an optional argument that specifies the host name. If you do not specify a host name, the program will use the host-name value that is set in your TCP/IP hosts file.

   Example:

   ```
   /WebSphere/V6R0/AppServer/bin> ivt.sh serverj default -p 9080 -host myhost
   ```

The IVT runs a series of verification tests and reports pass or fail status for each in the messages generated by the BBOWIVT job. The output is written to standard output and to the `WAS_HOME/profiles/default/logs/ivtClient.log` file.

Chapter 2. Configuring ports

When you configure WebSphere Application Server resources or assign port numbers to other applications, you must avoid conflicts with other assigned ports. In addition, you must explicitly enable access to particular port numbers when you configure a firewall.

1. Review the port number settings, especially when you are planning to coexist.
2. **Optional:** Change the port number settings.

   You can set port numbers when configuring (customizing) the product after installation. Start thinking about port numbers during the planning phase.

Chapter 3. Communicating with Web servers

The WebSphere Application Server 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 in WebSphere Application Server. The Web server plug-in uses the XML configuration file to determine whether a request is from the Web server or the Application Server.

- Install your Web server if it is not already installed.

  See the installation information provided with your Web server.
- Ensure that your Web server is configured to perform operations required by Web applications, such as GET and POST. Typically, this involves setting a directive in the Web server configuration file (such as the httpd.conf file for an IBM HTTP Server). Refer to the Web server documentation for instructions. If an operation is not enabled when a servlet or JSP file requiring the operation is accessed, an error message displays, such as this one from the IBM HTTP Server:

  ```
  IMW0093E Method POST is disabled on this server.
  ```
- Make sure the appropriate plug-in file has been installed on your Web server and the configureWeb_server_name script has been run to create and configure the Web server definition for this Web server.

  If you are using the IBM HTTP Server provided with the z/OS base operating system, see "Installing the WebSphere HTTP plug-in for z/OS" on page 133.

  If you are using a distributed platform Web server with a WebSphere Application Server running on a z/OS platform, FTP the plug-in to the Web server and use the Plug-in Installation wizard to install the appropriate plug-in file to your Web server. See the WebSphere Application Server Network Deployment version of the Information Center for a description of how to use this wizard.

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

1. 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. If you use the administrative console:

   a. Select the node that was created in the preceding step, and in the Server name field, enter the local name of the Web server for which you are creating a Web server definition.

   b. Use the wizard to complete the Web server definition.

   When creating a Web server definition for local or remote z/OS Web server in a WebSphere Application Server for z/OS cell, the process is slightly different. Use the Customization Dialog job BBOWCFGW to create a Web server definition in a stand-alone application server, and use Customization Dialog job BBODCFGW to create a Web server definition in a Network Deployment cell. Note that Web server definitions created in a stand-alone application server with BBOWCFGW are not federated into a Network Deployment cell as part of addNode.sh processing.
2. 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.
3. The master repository is updated and saved.

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

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 machine as the Web server, the regeneration usually takes about 30 to 60 seconds to complete. The regeneration takes longer if they are not both on the same machine.

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 or not 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

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

   a. Select **Servers > Web Servers >** *webserver* **> plug-in properties**.

   b. Select **Automatically generate plug-in configuration file** or click on one or more of the following topics to manually configure the plugin-cfg.xml file:
      - Caching
      - Request and response
      - Request routing
      - Service

      Web server plug-in configuration properties maps each property to one of these topics.

      Note: It is recommended that you do not manually update the plugin-cfg.xml file. Any manual updates you make for a given Web server are overridden whenever the plugin-cfg.xml file for that Web server is regenerated.

   c. Click **OK**.

   d. You might need to stop the application server and then start the application server again to enable the Web server to locate the plugin-cfg.xml file.

2. **Optional:** Edit the plug-in configuration file. You should not have to edit the configuration file. If you do edit this file remember that:

   - The file is in ASCII format (ISO-98859-1).

     Issue the following command to convert the file to EBCDIC format:

     ```
     > iconv  -f ISO8859-1 -t IBM-1047 plugin-cfg.xml.ASCII > plugin-cfg.xml.EBCDIC
     ```

     Edit the file, and then issue the following command to convert it back to ASCII format:

     ```
     > iconv  -f IBM-1047 -t ISO8859-1  plugin-cfg.xml.EBCDIC > plugin-cfg.xml.ASCII
     ```

   - Any manual changes you make to the file are overwritten the next time the file is regenerated.

3. If you want to enable the Application Server to use the private headers that the Web server plug-in sends, make sure the transport you are using is configured for SSL and is trusted. If a trust file definition is not included, the private headers will be ignored, and the application server might not locate the requested application.

   If you are using an HTTP transport, make sure the transport is configured for SSL and the TrustedProxy custom property for the transport is set to true.

   After you enable the use of private headers, this transport trusts all inbound private headers it receives. Therefore, you must ensure that all inbound paths to this transport are trusted.

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 is true:

   - The Web server is a local Web server. (It are located on the same machine as an application server.)

- The Web server is a remote IBM HTTP Server Version 6.0 that has a running IBM HTTP Server Administrative server.

If neither of these conditions is true, the plugin-cfg.xml file must be manually copied to the remote Web server's installation location.

Important: If you use the FTP function to perform the copy, and the configuration reload fails, check the file permissions on the plugin-cfg.xml file and make sure 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
```

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

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 on 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, you should consider enabling this service.

If you are making a series of simultaneous changes, like 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 elect **Servers** > **Application Servers** > *server_name* > **Administration Services >Web server plug-in configuration service** and then unselect the Enable automated Web server configuration processing option.

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

Web server plug-in properties settings

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** > **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.

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 set 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. Any server for which the host name can not be resolved is marked **unavailable** for the life of the configuration. No attempts to resolve the host name are made later on 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. When **false** is specified, DNS failures cause the Web server not to start.

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.

You can change the name of the plug-in configuration file. However, if you do change the file name, you must also change the Web server configuration to point to the new plug-in configuration file.

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 an IBM HTTP Server V6.1 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.

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
- 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 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 the z/OS platform, you can check the related messages in the TSO JOB log 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.

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.

Plug-in logging

Specifies the location and name of the http_plugin.log file. Also specifies the scope of messages in the log.

This field corresponds to the RequestMetrics traceLevel element in the plugin-cfg.xml file.

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.

Data type	String
Default	Error

Web server plug-in request and response optimization properties settings

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 > 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 20 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 IHSDisableNagle element in the plugin-cfg.xml file. It only appears 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 appears 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, 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.

By default, this field is not checked. Select this field to enable users to include content in POST, PUT, GET, and HEAD requests when a Content-Length or Transfer-encoding header is contained in the request header.

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 URI's for a sendRedirect.

This field corresponds to the AppServerPortPreference element in the plugin-cfg.xml file.

Specify:
- `webserverPort` if the port number from the host header of the HTTP request coming in is to be used.
- `hostHeader` if the port number on which the Web server received the request is to be used.

The default is `webserverPort`.

Web server plug-in caching properties settings

Use this page to view or change the caching properties for a Web server plug-in.

To view this administrative console page, click **Servers > Web Servers >** *Web_server_name* **Plug-in Properties > Caching Properties**.

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 not 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 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 settings

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 > Web Servers >** *Web_server_name* **Plug-in Properties > Plug-in** *server_cluster_name* **Properties**.

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 LoadBalanceWeight 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 ServerWaitforContinue 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 serverprocess 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 settings

Use this page to view or change the configuration settings for the Web server plug-in configuration service.

If you are using a stand-alone application server, click **Application Servers >** *server_name* > **Administration Services** > **Web server plug-in configuration service** to view this administrative console page.

If you are using the deployment manager, click **System Administration > Deployment manager > Administration Services > Web server plug-in configuration service**.

For the z/OS platform, the TSO JOB 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

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 **Application Servers >** *server_name*, and then under Additional Properties, 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

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. If `Set Timeout` is selected, 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 you select `No Timeout`, the plug-in uses blocked I/O to write requests to and read responses from the application server until the TCP connection times out.

This field is ignored for a plug-in running on a Solaris platform.

Data type	Integer
Default	No Timeout

Connect timeout

Specifies whether or not there is a limited amount of time the application server will maintain a connection with the Web server.

You can select either **No timeout** or **Set timeout**. If you select **Set timeout** you, must specify, in seconds, the length of time a connection with the Web server is to be maintained.

This property 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 whether or not the port is available. If no value is specified for this property, the plug-in performs a blocking connect in which the plug-in sits until an operating system times out (which could be 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 another application server defined for the requested application.

Data type	Integer
Default	0

Maximum number of connections that can be handled by the Application Server

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.

You can select either **No limit** or **Set limit**. If you select **Set limit** 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 attribute is set to either zero or -1, there is no limit to the number of pending connections to the Application Servers.

This attribute is ignored on the z/OS platform. The z/OS controller, working in conjunction with WLM, handles new connections dynamically.

Data type	Integer

Default -1

Use extended handshake to check whether application server is running

When selected, the Web server plug-in will use an extended handshake to check whether or not the Application Server is running.

This field corresponds to the ServerExtendedHandshake 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 other application servers.

If the plug-in performs some handshaking with the application server to ensure that it is started before it sends a request it can failover to another application server if it detects that the application server with which it is attempting to perform a handshake is down.

By default, this field is not checked. 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

This field corresponds to the WaitForContinue element in the plugin-cfg.xml file.

When selected, the Web server plug-in will send the header ″100 Continue″ to the application server before it sends the request content.

By default, this field is not checked. Select this field to enable this function.

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.

Table 1. Web server plug-in configuration properties

Administrative console panel	Field name	Configuration property name
In the administrative console, click **Servers** > **Web Servers** > *Web_server_name* > **Plug-in properties**	Refresh configuration interval	RefreshInterval
In the administrative console, click **Servers** > **Web Servers** > *Web_server_name* > **Plug-in properties**	Plug-in log file name	Log->name
In the administrative console, click **Servers** > **Web Servers** > *Web_server_name* > **Plug-in properties**	Plug-in logging	Log->LogLevel
In the administrative console, click **Servers** > **Web Servers** > *Web_server_name* > **Plug-in properties**	Ignore DNS failures during Web server startup	IgnoreDNSFailures

Table 1. Web server plug-in configuration properties (continued)

In the administrative console, click **Servers** > **Web Servers** > *Web_server_name* > **Plug-in properties**	KeyringLocation	Keyring
In the administrative console, click **Servers** > **Web Servers** > *Web_server_name* > **Plug-in properties**	StashfileLocation	Stashfile
In the administrative console, click **Servers** > **Web Servers** > *Web_server_name* > **Plug-in properties** > **Custom properties** > **New**	FIPSEnable	FIPSEnable
In the administrative console, click **Servers** > **Web Servers** > *Web_server_name* > **Plug-in properties** > **Request routing**	Load balancing option	LoadBalance
In the administrative console, click **Servers** > **Web Servers** > *Web_server_name* > **Plug-in properties** > **Request Routing**	Clone separator change	CloneSeparatorChange
In the administrative console, click **Servers** > **Web Servers** > *Web_server_name* > **Plug-in properties** > **Request Routing**	Retry interval	RetryInterval
In the administrative console, click **Servers** > *Web server_name* > **Plug-in properties** > **Request routing**	Maximum size of request content	PostSizeLimit
In the administrative console, click **Servers** > *Web server_name* > **Plug-in properties** > **Request routing**	Size of the buffer that is used to cache POST requests	PostBufferSize
In the administrative console, click **Servers** > **Web Servers** > *Web_server_name* > **Plug-in properties** > **Request routing**	Remove special headers	RemoveSpecialHeaders
In the administrative console, click **Application Servers** > *server_name* > **Web server plug-in properties**	Server role	PrimaryServers and BackupServers list
In the administrative console, click **Application Servers** > *server_name* > **Web server plug-in properties**	Connect timeout	Server ConnectTimeout
In the administrative console, click **Application Servers** > *server_name* > **Web server plug-in properties**	The read and write timeouts for all the connections to the application server	ServerIOTimeout
In the administrative console, click **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 **Application Servers** > *server_name* > **Web server plug-in properties**	Send the header ″100 Continue″ before sending the request content	WaitForContinue

Table 1. Web server plug-in configuration properties (continued)

In the administrative console, click **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 > 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 > Web Servers >** *Web_server_name* **> Plug-in properties > Request and Response**	Enable Nagle Algorithm for the IIS Web Server	IISDisableNagle
In the administrative console, click **Servers > Web Servers >** *Web_server_name* **> Plug-in properties > Request and Response**	Virtual host matching	VHostMatchingCompat
In the administrative console, click **Servers > 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 > Web Servers >** *Web_server_name* **> Plug-in properties > Request and Response**	Accept content for all requests	AcceptAllContent
In the administrative console, click **Servers > Web Servers >** *Web_server_name* **> Plug-in properties > Request and Response**	Chunk HTTP response to the client	ChunkedResponse
In the administrative console, click **Servers > 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 > 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 > Web Servers >** *Web_server_name* **> Plug-in properties > Caching**	Maximum cache size	ESIMaxCacheSize

Table 1. Web server plug-in configuration properties (continued)

In the administrative console, click **Servers** > **Web Servers** > *Web_server_name* > **Plug-in properties** > **Caching**	Enable invalidation monitor to receive notifications	ESIInvalidationMonitor

Web server plug-in connections

The WebSphere Application Server 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 HTTP inbound channel's **Maximum persistent requests** property.)
- 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.

Important: 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 third-party 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 application's call to getRemoteUser() 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.

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.

If an application's call to getRemoteUser() returns a null value, or if the correct remote user information is not being added to the Web server plug-in's data structure, make sure the remote user parameter within the WebAgent is still set to **YES**. (Sometimes this parameter gets set to **NO** when service is applied.)

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 Web site 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.

Installing the WebSphere HTTP plug-in for z/OS

The WebSphere HTTP plug-in for z/OS ships as part of WebSphere Application Server for the z/OS platform. To use this plug-in, you must have a Version 5.3 IBM HTTP Server for z/OS or OS/390 configured as part of a z/OS system.

After the application server, the HTTP Server and the plug-in are properly configured:
- WebSphere Application Server for the z/OS platform can use this plug-in to perform regular plug-in functions.
- Requests can be routed from a browser, through the HTTP Server and plug-in, to an application server on which the requested application is deployed. (New requests are sent to randomly selected application servers on which this application is deployed. After a session is established, requests get routed back to the application server assigned to the original request.)
- "Private headers" on page 140 can be used as a mechanism for forwarding proxy information from the Web server plug-in to an application server on a z/OS system. (This information is not otherwise included with HTTP requests.)

1. Make sure a Version 5.3 IBM HTTP Server for z/OS and OS/390 is installed on a z/OS or OS/390 system.
2. If the HTTP Server is not installed on the same LPAR as the application server, perform the following steps to download, in binary format, the WebSphere HTTP plug-in for z/OS and the plugin-cfg.xml file from the LPAR on which the application server is running to the LPAR where the HTTP Server is installed. The directory into which the plug-in is downloaded must be readable to the MVS ID under which the HTTP Server is executing.
   a. Use FTP or another file transfer mechanism to download, in binary format, the WebSphere HTTP plug-in for z/OS from your WebSphere Application Server for z/OS system to the HTTP Server's system and into a directory that is readable to the MVS ID under which the HTTP Server is executing. The ihs390WAS61Plugin_http.so DLL is located in the WebSphere Application Server *app_server_root*/bin/ directory
   b. Set the permissions (755 +p) on the plug-in's ihs390WAS61Plugin_http.so file. Using an authorized z/OS user ID, issue the following commands from an OMVS command line prompt to turn on the ″p″ bit in the HFS where the WebSphere HTTP plug-in for z/OS is now located:

      ```
      chmod 777 ihs390WAS61Plugin_http.so
      extattr +p ihs390WAS61Plugin_http.so
      ```
3. Add ServerInit, ServerTerm, and Service directives to the httpd.conf configuration file of the HTTP Server:
   - Add the following ServerInit and ServerTerm directives to indicate the entry points to the plug-in's initialization and exit routines. These routines exist as entry points init_exit, and term_exit, respectively, within the ihs390WAS61Plugin_http.so DLL file.

     Important:
     – In this discussion, the ServerInit and Service directives are split for printing purposes. In the actual httpd.conf file, enter each of these directives on a single line.
     – In the ServerInit directive, *http_plugin_conf* represents the full path to the location of the plugin-cfg.xml file.
     – In the ServerInit, Service and ServerTerm directives, *http_plugin_dir* represents either:
       a. The full path of the Application Server's *app_server_root*/bin/ path, if the HTTP Server is on the same LPAR as the WebSphere Application Server, or
       b. The full path that you designated as the destination of the FTP operation in the previous step, if the HTTP Server is not on the same LPAR as the WebSphere Application Server.

     ```
     ServerInit /http_plugin_dir/bin/
         ihs390WAS61Plugin_http.so:init_exit /http_plugin_conf/
         plugin-cfg.xml
     ServerTerm /http_plugin_dir/bin/ihs390WAS61Plugin_http.so:term_exit
     ```
   - Add the following Service directive for each application that will be using the Web server plug-in. This directive indicates the entry point to the plug-in's request routine. The request routine exists as the entry point service_exit within the ihs390WAS61Plugin_http.so Dynamic Link Library (DLL) file.

     ```
     Service /webapp_contextroot/*  /http_plugin_dir/
          ihs390WAS61Plugin_http.so:service_exit
     ```

     webapp_contextroot is the application's context root

   Notes:
   a. The HTTP Server interprets a blank in a directive specification as a delimiter and a number sign (#) as the beginning of a comment that should be ignored. Therefore, if you need to use a blank or number sign in a directive, you must include a backslash (\) before the blank or number sign to enable the HTTP Server to correctly process the directive.
   b. If a servlet sets an HTTP response code by any means, such as using methods lastModified() or setStatus(), and the client does not receive the expected response code, add the following directive to the HTTP Server configuration file:

      ```
      ServiceSync On
      ```

c. If you want to use Secure-socket layer (SSL), make sure that the HTTP Server is configured for SSL. The SSL connection between the Web server plug-in and the J2EE server uses the SSL session established by the HTTP Server. (See *z/OS HTTP Server Planning, Installing, and Using, Version 5.3*, SC34-4826, for a description of how to configure the HTTP Server for SSL.)

4. Make sure the httpd.conf file does not contain a Pass directive that is set to **/***.. If the httpd.conf file contains the line:

```
Pass      /*
```

Place a # (number sign) in the first column of that line to comment it out.

5. Create a Web server definition for the local or remote z/OS Web server using the appropriate Customization Dialog panel.

6. Configure the plug-in. Use either the administrative console or issue the genplugincfg command to create your plugin-cfg.xml file.

Note: Both methods create the plug-in configuration file, plugin-cfg.xml, in ASCII format. (Previously, the configuration file was generated in EBCDIC format.)

a. If you need to edit this file, issue the following command to convert the file to EBCDIC format:

```
> iconv  -f ISO8859-1 -t IBM-1047 plugin-cfg.xml.ASCII > plugin-cfg.xml.EBCDIC
```

b. Edit the file, and then issue the following command to convert it back to ASCII format:

```
> iconv  -f IBM-1047 -t ISO8859-1  plugin-cfg.xml.EBCDIC > plugin-cfg.xml.ASCII
```

To use the administrative console:

a. Select **Servers > Web Servers >** *webserver* **> plug-in properties**.

b. Select **Automatically generate plug-in configuration file** or click on one or more of the following topics to manually configure the plugin-cfg.xml file:
- Caching
- Request and response
- Request routing
- Service

c. Click **OK**.

d. You might need to stop the application server and then start the application server again to enable the Web server to locate the plugin-cfg.xml file.

7. Make sure the virtual host is configured with an alias for the port number used by the z/OS V5.3 HTTP Server. If you manually configured the plugin-cfg.xml file, go to the **Servers > Application Servers > plug-in properties > Request routing** page in the administrative console and make sure that **Physically using the port specified in request** is selected for **Virtual host matching**.

8. If you want to enable the Web server plug-in to use private headers, define an SSL configuration repertoire that defines a trust file. Then in the administrative console, select **Application servers > server1 > Web Container Settings > Web Container Transport Chains >** *transport_chain* > **SSL Inbound Channel (SSL_2)** and specify this repertoire for that transport chain. If you try to use private headers without setting up an SSL configuration repertoire that does not include a trust file definition, the private headers will be ignored. If the private headers are ignored, the application server might not locate the requested application.

After you enable the use of private headers, the transport chain's SSL inbound channel trusts all private headers it receives. Therefore, you must ensure that all paths to the transport chain's SSL inbound channel are trusted.

9. Stop the application server and the HTTP Server and start them again.

The configuration is complete. To activate the configuration, stop and restart both the application server and the HTTP Server. If the WebSphere HTTP plug-in for z/OS comes up when the HTTP Server starts again, you receive the following messages:

```
WebSphere HTTP plug-in for z/OS Version 6.00 Service Level 0 is starting
 WebSphere HTTP plug-in for z/OS initializing with configuration file :
        fully_qualified_path_to_the_plugin-cfg.xml_file
WebSphere HTTP plug-in for z/OS initialization went OK :-)
```

Setting up communication between a z/OS Application Server and a Web server running on a workstation

If you have WebSphere Application Server installed on your z/OS system, you can configure an application server that is running on that system to communicate with a Web server and Web server plug-in for WebSphere Application Server that is running on a distributed platform, such as Linux or Microsoft Windows.

This configuration enables requests for a particular application to be routed from a browser, through the Web server and Web server plug-in, to one of the application servers defined for that application on the z/OS system. (An application is associated with a Web server when it is deployed on an application server.)

In addition to regular plug-in functions, Web server plug-ins for the WebSphere Application Server uses "Private headers" on page 140 as a mechanism for forwarding proxy information from the plug-ins to an application server running on the z/OS platform. This information is not otherwise included with HTTP requests.

Before you begin:

The WebSphere Application Server for z/OS product package contains the Web server plug-ins for Web servers that run on distributed platforms and are supported by the WebSphere Application Server Version 6 products.

1. Go to the WebSphere Application Server administrative console that is running on the z/OS platform and make sure the virtual host contains an alias for the port number used by the Web server. Specify this same port on a <Virtual Hostname> element in the plug-in plugin-cfg.xml file.
2. Generate a plug-in configuration file. The plug-in configuration file that is created when you run the Plug-in installation wizard does not include any information about the a/OS applications for which it will be receiving requests. Therefore, you must use the WebSphere Application Server administrative console located on your z/OS system to generate a plug-in configuration file that includes this application information.

   To generate a plug-in configuration file:

   a. Select **Servers > Web Servers >** *web_server* **> plug-in properties**.
   b. Select **Automatically generate plug-in configuration file** or click on one or more of the following topics to manually configure the plugin-cfg.xml file:
      - Caching
      - Request and response
      - Request routing
      - Service
   c. Click **OK**.
3. **Optional:** Make any additional changes to the plug-in configuration file. Usually, you will not have to make any manual changes to the plug-in configuration file you just created. However if you do need to make changes remember that these changes will be overwritten the next time the configuration file is regenerated.

   Starting with Version V6.0.1, the plug-in configuration file, plugin-cfg.xml, is generated in ASCII format. (Previously, the configuration file was generated in EBCDIC format.) If you need to edit this file, issue the following command to convert the file to EBCDIC format:

   ```
   > iconv  -f ISO8859-1 -t IBM-1047 plugin-cfg.xml.ASCII > plugin-cfg.xml.EBCDIC
   ```

Edit the file, and then issue the following command to convert it back to ASCII format:

```
> iconv  -f IBM-1047 -t ISO8859-1  plugin-cfg.xml.EBCDIC > plugin-cfg.xml.ASCII
```

4. **Optional:** To enable the Application Server to use the private headers that the Web server plug-in sends, make sure the transport you are using is configured for SSL and is trusted. If your transport is a transport chain you must define security for that chain that includes a trust file definition. If a trust file definition is not included, the private headers will be ignored, and the application server might not locate the requested application.

   After you enable the use of private headers, this transport trusts all inbound private headers it receives. Therefore, you must ensure that all inbound paths to this transport are trusted.
5. If you want to use Secure-socket layer (SSL) with this configuration, use the plug-in's installation wizard to install the appropriate GSKIT installation image file on your workstation.
6. Download the newly generated plug-in configuration file to the Web server. You must replace the plug-in configuration file the Plug-in installation wizard created with the one you just generated on your z/OS system. Therefore, after you finish generating the plug-in configuration file, download it to the directory on the Web server that contains the plug-in configuration file that the Plug-in installation wizard generated.

The configuration is complete. To activate the configuration, stop and restart both the WebSphere Application Server that is running on your z/OS system , and the Web server that is running on your workstation.

Checking your IBM HTTP Server version

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

1. Change the directory to the installation root of the Web server.
2. Find the subdirectory that contains the executable. The executable is:
   - apachectl
3. Issue the command.

   Use a JCL procedure or issue a command from the OMVS command line to start a z/OS HTTP Server as a started task.

If you use a JCL procedure to start the z/OS HTTP Server, a banner similar to the following banner will appear in the Job Log. If you issued a command from the OMVS command line to start the HTTP Server, this banner will follow the command on the OMVS screen:

```
 ........... This is IBM HTTP Server V5R3M0

............ Built on Nov 26 2005 at 12:08:21.
............ Started at Mon Mar 28 18:32:33 2005
............ Running as "WEBSRV", UID:0, GID:205
```

In this example the version for this HTTP Server for z/OS is v5.3.0. If you need to contact IBM Support about a problem with your z/OS HTTP Sever, use this banner to determine the ″Built on date″ for your HTTP Server and include this information in your problem report.

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 %was_profile_home%/config/cells directory.

You must update the global Web server plug-in configuration file whenever you:

- 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:

1. Click **Environment > Update global Web server plug-in configuration**.
2. Click **OK** to update the plugin-cfg.xml file.
3.
4. 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.

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.

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 the base WebSphere Application Server installation, 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. The plugin-cfg.xml file is located in the application server /config/cells directory.

- **Command line**:

At a command prompt, enter the following command to change to 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 *WAS_HOME*/AppServer directories.

```
GenPluginCfg.sh -destination.root WAS_HOME/AppServer
```

For example, issue the following command from the DeploymentManager/bin directory.

```
GenPluginCfg -destination.root "/WebSphere/V5R0M0/AppServer"
```

- **plugin-cfg.xml file**:

  Edit the plugin-cfg.xml file, located in the *app_server_root*/DeploymentManager/config/cells directory, 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 WebSphere Application Server 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.

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.

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. Use the following table to assist you in selecting the correct GSKIT installation image file.

Operating system	GSKit 7 Installation image file
Windows	No image name
AIX	gskta.rte
HP-UX	gsk7bas

Solaris Operating Environment	gsk7bas
Linux	gsk7bas_7.0.3.1.i386.rpm
Linux390	gsk7bas-7.0.3.1.s390.rpm
LinuxPPC	gsk7bas-7.0.3.1.ppc.rpm

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

- IBM HTTP Server documentation at http://www-3.ibm.com/software/webservers/httpservers/library.html
- 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

Web server plug-in tuning tips

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.

The WebSphere Application Server uses the z/OS native Workload Management (WLM) functionality to dynamically balance the workload of application servers defined to a z/OS HTTP Server. See the z/OS publication *IBM HTTP Server Planning, Installing and Using v5.3* for more information.

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.

plugin-cfg.xml file

The plugin-cfg.xml file includes the following elements and attributes. Unless indicated otherwise, each element and attribute can only be specified once within the plugin-cfg.xml file.

Starting with Version V6.0.1, the plug-in configuration file is generated in ASCII format (ISO-98859-1). (Previously the configuration file was generated in EBCDIC format.) If you need to edit this file, issue the following command to convert the file to EBCDIC format:

```
> iconv  -f ISO8859-1 -t IBM-1047 plugin-cfg.xml.ASCII > plugin-cfg.xml.EBCDIC
```

Edit the file, and then issue the following command to convert it back to ASCII format:

```
> iconv  -f IBM-1047 -t ISO8859-1  plugin-cfg.xml.EBCDIC > plugin-cfg.xml.ASCII
```

CAUTION:
Use the administrative console to set these properties for a given Web server definition. Any manual changes you make to the plug-in configuration file for a given Web server are overridden whenever the file is regenerated.

Config (required)

This element starts the WebSphere HTTP plug-in configuration file. It can include one or more of the following elements and attributes.

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 ServerCluster is able to resolve the host name. Any server for which the host name can not be resolved is marked *unavailable* for the life of the configuration. No attempts to resolve the host name are made later on 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

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 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.

ASDisableNagle

Specifies whether the user wants to disable nagle algorithm for the connection between the plug-in and the application server. By default, nagle algorithm is enabled.

The value can be true or false.

IISDisableNagle

Specifies whether the user wants to disable nagle algorithm on Microsoft Internet Informations Services (IIS). By default, nagle algorithm is enabled.

The value can be true or false.

AppServerPortPreference

This attribute is used to specify which port number the Application Server should use to build URI's for a sendRedirect. The following values can be specified:

- webserverPort if the port number from the host header of the HTTP request coming in is to be used.
- hostHeader if the port number on which the Web server received the request is to be used.

The default is hostHeader.

ResponseChunkSize

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.

The ResponseChunkSize attribute lets you specify the maximum chunk size to use when reading the response body. For example, Config ResponseChunkSize=″N″>, where N equals the chunk size in kilobytes.

If the content length of the response body is unknown, a buffer size of N kilobytes is allocated and the body is read in N kilobyte 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 64k.

AcceptAllContent

Specifies whether or not 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 to be expected and read for all requests
- False if content only is only to be expected and read for POST and PUT requests.

False is the default.

ChunkedResponse

Specifies whether the plug-in should chunk the response to the client when a Transfer-Encoding : Chunked response header is present in the response.

This attribute only applies to the IIS, IPlanet, and Domino Web servers. 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.

false is the default.

IISPluginPriority

Specifies the priority in which the IIS Web server loads the WebSphere Web server plug-in. You can specify one of the following values for this attribute:

- High
- Medium
- Low

The default value is High.

NOTES:

- The IIS Web server uses this value during startup. Therefore, the Web server must be restarted before this change will take effect.
- 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/extensions. If problems occur while using a priority of Medium or Low, you will have to rearrange the order or change the priority of the interfering filter/extension.

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.

For example, you might specify the following:

```
<Log LogLevel="Error" Name="/log_directory/filename"/>
```

Name (exactly one attribute for each Log)

The fully qualified path to the log file to which the plug-in will write error messages.

Note: The time the information was written to the log file and the process ID are appended to the file name specified on this element.

LogLevel (zero or one attribute for each Log)

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 LogLevel is not specified for the Log element, 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 file system to fill up very quickly. A Trace setting should never be used in a normally functioning environment as it adversely affects performance.

Property Name=″esiEnable″ Value=″*true/false*″

Used to enable or disable the Edge Side Include (ESI) processor. If the ESI processor is disabled, the other ESI elements in this file are ignored.

Value can be set to true or false. By default, the ESI processor is enabled (set to true).

Property Name=″esiMaxCacheSize″ Value=″*interger*″

An integer specifying, 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.

Property Name=″ESIInvalidationMonitor″ Value=″*true/false*″

Used to indicate whether or not the ESI processor should receive invalidations from the Application Server.

Value can be set to true or false. By default, this property is set to false.

This property must always be set to false for the WebSphere for z/OS HTTP Server plug-in.

Property Name=″FIPSEnable″ Value=″*true/false*″

Used to indicate whether or not the Federal Information Processing Standard (FIPS) is enabled for making secure (SSL) connections to the Application Server. This property should be set to true, if FIPS is enabled on the Application Server..

Value can be set to true or false. By default, this property is set to false.

ServerCluster (one or more elements for each Config)

A group of servers that are generally configured to service the same types of requests.

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 will load balance across the defined servers using either a Round Robin or a Random algorithm. The default is Round Robin.

Following is an example of a ServerCluster element

```
<ServerCluster CloneSeparatorChange="false"
        LoadBalance="Round Robin" Name="Cluster1"
        PostSizeLimit="10000000" RemoveSpecialHeaders="true"
        RetryInterval="60">
<Server
CloneID="BA36BEC1EB243D8B000000E4000000030926301B"
            ConnectTimeout="0" ExtendedHandshake="false"
            LoadBalanceWeight="2" MaxConnections="0"
            Name="SY1_ClusterMember1" WaitForContinue="false">
<Transport Hostname="BOSSXXXX.PLEX1.L2.IBM.COM" Port="9084" Protocol="http"/>
<Transport Hostname="BOSSXXXX.PLEX1.L2.IBM.COM" Port="0" Protocol="https">
<Property Name="Keyring" value="/WebSphere/V6R0M0/DeploymentManager/etc/
            plugin-key.kdb"/>
<Property Name="Stashfile" value=""/WebSphere/V6R0M0/DeploymentManager/etc/
            plugin-key.sth"/>
<Property Name="certLabel" Value="selfsigned"/>
</Transport>
</Server>
<Server CloneID="BA36BED017FDF40E000000E4000000030926301B"
            ConnectTimeout="0" ExtendedHandshake="false"
            LoadBalanceWeight="2" MaxConnections="0"
            Name="SY1_ClusterMember2" WaitForContinue="false">
<Transport Hostname="BOSSXXXX.PLEX1.L2.IBM.COM" Port="9085" Protocol="http"/>
<Transport Hostname="BOSSXXXX.PLEX1.L2.IBM.COM" Port="0" Protocol="https">
<Property Name="Keyring" value="/WebSphere/V6R0M0/DeploymentManager/etc/
            plugin-key.kdb"/
<Property Name="Stashfile" value="/WebSphere/V6R0M0/DeploymentManager/etc/
            plugin-key.sth"/>
<Property Name="certLabel" Value="selfsigned"/>
</Transport>
</Server>
<PrimaryServers>
<Server Name="Server Name="SY1_ClusterMember1"/>
<Server Name="Server Name="SY1_ClusterMember2"/>
</PrimaryServers>
</ServerCluster>
```

Note: If you are using the WebSphere HTTP Plug-in for z/OS, the Property Name=keyring and the Property Name=stashfile elements included here will be ignored if they are included in the plugin-cfg.xml file for that plug-in. The WebSphere HTTP Plug-in for z/OS uses the SSL setup specified in the hosting HTTP Server's httpd.conf file and does not look for these elements in the plugin-cfg.xml file.

Name (exactly one attribute for each ServerCluster)

The logical or administrative name to be used for this group of servers.

LoadBalance (zero or one attribute for each ServerCluster)

The default load balancing type is Round Robin.

The Round Robin implementation has a random starting point. The first server will be picked randomly. Round Robin will be used to pick 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.

RetryInterval (zero or one attribute for each ServerCluster)

An integer specifying the length of time that should elapse from the time that a server is marked down to the time that the plug-in will retry a connection. The default is 60 seconds.

RemoveSpecialHeaders (zero or one attribute for each ServerCluster)

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. By default the plug-in will remove these headers from incoming requests before adding the headers it is supposed to add.

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 (zero or one attribute for each ServerCluster)

Some pervasive devices cannot handle the colon character (:) used to separate clone IDs in conjunction with session affinity. This attribute for the server group tells the plug-in to expect the plus character (+) as the clone separator. You must change application server configurations so that an application server separates clone IDs with the plus character as well.

The value can be true or false.

PostSizeLimit (zero or one attribute for each ServerCluster)

The maximum number of bytes of request content allowed in order 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 bytes, which indicates that there is no limit for the post size.

Server (one or more elements for each ServerCluster)

A WebSphere Application Server instance that is configured to handle requests routed to it given the routing rules of the plug-in configuration. The Server should correspond to an application server running on either the local machine or a remote machine.

Name (exactly one attribute for each Server)

The administrative or logical name for the server.

CloneID (zero or one attribute for each Server)

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.

This attribute is used in conjunction 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.

If you are not using session affinity then it is best to remove these clone IDs from the configuration because there is added request processing in the plug-in when these are set. If clone IDs are not in the plug-in then it is assumed that session affinity is not on and the request is load balanced across the cluster.

WaitForContinue (zero or one attribute for each Server)

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.

This property will be ignored for POST requests in order to prevent a failure from occurring if the Application server closes a connection because of a keep alive time-out.

Enable this function (set to true) when configuring the plug-in to work with certain types of proxy firewalls.

LoadBalanceWeight (zero or one attribute for each Server)

Specifies the weight associated with this server when the plug-in does weighted Round Robin load balancing. The starting value for a server can be any integer between 0 and 20. However, zero should be specified only for a server that is shut down.

The algorithm for this attribute decrements all weights within the server cluster until all weights reach zero. After the weight specified for a particular server reaches zero, no more requests are routed to that server until all servers in the cluster have a weight of zero. After all servers reach zero, the weights for all servers in the cluster are reset and the algorithm starts over.

When a server is shut down, it is recommended that you 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 (zero or one attribute for each Server)

The ConnectTimeout attribute of a Server element 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.

If no ConnectTimeout value is 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.

ExtendedHandshake (zero or one attribute for each Server)

The ExtendedHandshake attribute 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.

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 other application servers.

The plug-in performs some handshaking with the application server to ensure that it is started before sending the request. This enables the plug-in to failover in the event the application server is down.

The value can be true or false.

MaxConnections (one element for each Server)

The MaxConnections attribute is used to specify the maximum number of pending connections to an Application Server that can be flowing through a Web server process at any point in time.

For example, assuming that:

- The application server is fronted by 5 nodes that are running an IBM HTTP Server.
- Each node starts 2 processes.
- The MaxConnections attribute 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 the MaxConnections attribute, 50, for a total of 500 connections.)

This attribute is not necessary on the z/OS platform. The z/OS controller working in conjunction with WLM, handles new connections dynamically.

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 (one or more elements for each Server)

The transport for reading and writing requests to a particular WebSphere application server instance. The transport provides the information needed to determine the location of the application server to which the request will be sent. If the Server has multiple transports defined to use the same protocol, the first one will be used.

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 will be performed to determine the appropriate transport to use to send the request to the application server.

Hostname (exactly one attribute for each Transport)

The host name or IP address of the machine on which the WebSphere application server instance is running.

Port (exactly one attribute for each Transport)

The port on which the WebSphere application server instance is listening.

Protocol (exactly one attribute for each Transport)

The protocol to use when communicating over this transport -- either HTTP or HTTPS.

Property (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:

```
<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"/>
```

Note: The default password for viewing the plugin-key.kdb file using iKeyMan is WebAS.

Name (exactly one attribute for each Property)

The name of the Property being defined. Supported names recognized by the transport are keyring, stashfile, and password.

Note: password is the only name that can be specified for the WebSphere HTTP Plug-in for z/OS. keyring, and stashfile, if specified, will be ignored.

Value (exactly one attribute for each Property)

The value of the Property being defined.

ServerIOTimeout

The ServerIOTimeout attribute of a server element enables the plug-in to set a time out value, in seconds, for sending requests to and reading responses from the application server. 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, if you specify:

```
<Server Name="server1" ServerIOTimeout=300>
```

In this case, 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 time out 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 a couple of minutes for an application server to process a request. Setting the value of the ServerIOTimeout attribute too low could cause the plug-in to send a false server error response to the client.

ClusterAddress (zero or one element for each ServerCluster)

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 only define 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.

Important: 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.

PrimaryServers (zero or one element for each server cluster)

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.

BackupServers (zero or one element for each server cluster)

Specifies a list of servers to which requests should be sent to if all servers 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 received from an application server.

VirtualHostGroup

A group of virtual host names that will be specified in the HTTP Host header. Enables you to group virtual host definitions together that are configured to handle similar types of requests.

Following is an example of a VirtualHost Group 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 (exactly one attribute for each VirtualHostGroup)

The logical or administrative name to be used for this group of virtual hosts.

VirtualHost (one or more elements for each VirtualHostGroup)

The name used for a virtual or real machine used to determine if incoming requests should be handled by WebSphere Application Server or not. Use this element to specify host names that will be in the HTTP Host header which should be seen for requests that need to be handled by the application server. You can specify specific host names and ports that incoming requests will have or specify an asterisk (*) for either the host name, port, or both.

Name (exactly one attribute for each VirtualHost)

The actual name that should be specified in the HTTP Host header in order to match successfully with this 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 what those combinations are.

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 will match this rule. If no port is specified in the definition the default HTTP port of 80 is assumed.

UriGroup

A group of URIs that will be specified on the HTTP request line. The same application server must be able to handle the URIs. The route will compare the incoming URI with the URIs in the group to determine if the application server will handle the request.

Following is an example of a UriGroup element and associated elements and attributes:

```
<UriGroup Name="Uris">
<Uri Name="/servlet/snoop"/>
<Uri Name="/webapp/*"/>
<Uri Name="*.jsp"/>
</UriGroup>
```

Name (exactly one attribute for each UriGroup)

The logical or administrative name for this group of URIs.

Uri (one or more elements for each UriGroup)

The virtual path to the resource that will be serviced by WebSphere Application Server. Each URI specifies the incoming URLs that need to be handled by the application server. You can use a wildcard in these definitions.

Name (exactly one attribute for each Uri)

The actual string that should be specified in the HTTP request line in order 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 a URI having <Uri Name=″*Web_application_URI*/servlet/*″> is generated.

AffinityCookie (zero or one attribute for each Uri)

The name of the cookie the plug-in should use 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.

AffinityURLIdentifier (zero or one attribute for each Uri)

The name of the identifier the plug-in should use 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.

Route A request routing rule by which the plug-in will determine if an incoming request should be handled by a WebSphere application server.

The route definition is the central element of the plug-in configuration. It specifies how the plug-in will handle 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 should be sent on to the ServerCluster defined in this route.

Following is an example of this element:

```
<Route VirtualHostGroup="Hosts" UriGroup="Uris" ServerCluster="servers/>
```

VirtualHostGroup (zero or one attribute for each Route)
The group of virtual hosts that should be used in route determination. The incoming host header and server port are matched to determine if this request should be handled by the application server.

It is possible to omit this from the route definition. If it is not present then every request will match during the virtual host match portion of route determination.

UriGroup (zero or one attribute for each Route)
The group of URIs to use for determining the route. The incoming URI for the request is matched to the defined URIs in this group to determine if this request should be handled by the application server.

It is possible to omit this from the route definition. If it is not present than every request will match during the URI match portion of route determination.

ServerCluster (exactly one attribute for each Route)
The cluster to which to send request that successfully match the route.

The cluster that should be 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 defined within this cluster.

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.

Important: Non-IBM HTTP Server Web servers must reside on a managed node to facilitate plug-in administration functions and generation and propagation of the `plugin-cfg.xml` file.

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.

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

1. Install your WebSphere Application Server product.
2. Install IBM HTTP Server or another supported Web server.
3. Install the binary plug-in module using the Plug-ins installation wizard.
4. 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.

Select one of the following options:

- **Using the administrative console.** You must create a Web server definition on an existing application server or unmanaged node.
  a. Click **Servers > Web servers > New** and use the **Create new Web server entry** wizard to create the Web server definition.
  b. Select the appropriate node.
  c. 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 `IBMHTTPServer6.0`.
     – 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.
  d. Select a template. Select a system template or a user-defined template for the Web server you want to create.
  e. Confirmation of Web server creation.
- **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.

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:

- 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

This topic describes how to 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 one machine and the application server is on another.

Important: Non-IBM HTTP Server Web servers must reside on a managed node to facilitate plug-in administration functions and generation and propagation of the `plugin-cfg.xml` file.

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 a firewall. You must create a remote Web server on an unmanaged node. Unmanaged nodes are nodes without node agents. Since 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. In this case, 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.

There is no IBM HTTP Server administration server on z/OS platforms.

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

1. Install your WebSphere Application Server product.
2. Install IBM HTTP Server or another supported Web server.
3. Install the binary plug-in module using the Plug-ins installation wizard.
4. 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 > Web servers > New** to launch the **Create new Web server entry** wizard. You will create the new Web server definition using this wizard. The wizard 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 `IBMHTTPServer6.0`.
           – **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.

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.

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

5. Run the `setupadm` script on Linux and UNIX platforms. 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 Setting permissions manually for instructions.

6. Run the `htpasswd` script on Linux, UNIX and Windows platforms. 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 Linux and UNIX 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.

7. Start IBM HTTP Server. Refer to Starting the IBM HTTP administration server on Windows operating systems or Starting the IBM HTTP administration server on Linux and UNIX platforms for instructions.

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 installaion 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:

- 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.

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 Web server properties, for example, installation root, port, configuration file paths, and 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.

You have 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 an Express or Base 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 or dummy 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 and where a node agent is present. This node can exist in an Express, Base, or deployment manager environment. A dummy 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.

Administration of the Web server

The IBM HTTP Server Web server is administered and managed on the Web server collection panel in the WebSphere Application Server administrative console. You can start and stop IBM HTTP Server from the administrative console.

In WebSphere Application Server V6.1, you can create a Web server on a stand-alone node, enabling the Web server definition created on the stand-alone node to be included in the Deployment Manager's managed node when it is federated. Web servers that are defined on a stand-alone node are managed just as a Web server that is defined on an unmanaged node. An IBM HTTP Server Web server that is defined on a stand-alone node is managed by the IBM HTTP Server administration server. Non-IBM HTTP Server Web servers are not managed because no administrative agent exists to handle administration management.

Editing the Web server type

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

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.

1. From the WebSphere Application Server administrative console, click **Servers > 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.

   A new Web server type for WebSphere Application Server for z/OS exists: HTTPSERVER_ZOS.
4. Click **Apply**.

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

Web server collection

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

Web servers

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

To create a new Web server, click the **New** button 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.
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 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 the name of the node on which the Web server is defined.
Version	Specifies the version of the WebSphere Application Server node 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 will need to start the IBM HTTP Server administration server before you can start and stop IBM HTTP Server.

Note that 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.

Web server configuration

Use this page to configure Web server properties.

Web servers

To view this administrative console page click **Servers > 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 - Windows operating systems only
Specifies the 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.

Web server log file

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

To view this administrative console page, click **Servers > 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 > 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.

Remote Web server management

Use this page to configure the properties of an IBM HTTP Server Web server that is created on an unmanaged node.

Create a new, unmanaged node by clicking **System administration > Nodes > Add Node**. Create a Web server using the newly-created, unmanaged node by clicking **Servers > Web Servers > New**. To view the administrative console page for Remote Web server management, click **Servers > Web Servers >** *Web_server_name* > **Remote Web server management**.

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.

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 > Web Servers >** *Web_server_name* > **Configuration settings > Global directives**.

Security enabled

Specifies if security is enabled in your Web server.

Server name

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

Listen port

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

Document root

The directory where the Web server will serve files.

Keystore filename

Specifies the name you have assigned to your keystore.

Keystore directory

Specifies the target directory of your keystore on the machine where the Web server is installed.

SSL Version 2 timeout

Specifies the SSL Version 2 timeout.

SSL Version 3 timeout

Specifies the SSL Version 3 timeout.

Keystore certificate label

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

Virtual hosts collection

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

To view this administrative console page, click **Servers > 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**.

Virtual hosts detail

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

To view this administrative console page, click **Servers > 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.

Web server definition

IBM HTTP Server for OS/390 and z/OS does not support remote administration. For WebSphere Application Server for z/OS, use one of the following methods to manage Web server customization files:

- **Define a Web server definition in a managed node.**

  This method provides the most function, but requires that the Web server configuration files be stored in a read and write directory that is accessible to a managed (application server) node in the WebSphere Application Server for z/OS cell. Create the managed node with the Customization Dialog, either directly or by federating a stand-alone application server; then use the administrative console to create the Web server definition.

When a Web server definition is defined in a managed node, you can regenerate the `plugin-cfg.xml` configuration file directly to the Web server configuration directory location. You can then retrieve, edit, and replace the `httpd.conf` file that controls Web server operation.

- **Define a Web server definition in an unmanaged node.**

  Use this method when a Web server is defined in a stand-alone application server, or on a z/OS system that does not share read and write directories with a WebSphere Application Server for z/OS managed node.

  Use the Customization Dialog job BBOWCFGW (stand-alone application server) or BBODCFGW (for a Network Deployment cell) to create an unmanaged node and a Web server definition.

  When a Web server definition is defined in an unmanaged node, you can regenerate the plug-in configuration file, but cannot edit the `httpd.conf` file. If the Web server does not share a read and write configuration directory with the z/OS system running the stand-alone application server or deployment manager, you are responsible for moving the `plugin-conf.xml` file to the z/OS system for the Web server.

Chapter 4. Setting up the administrative architecture

You can monitor and control incorporated nodes and the resources on those nodes by using these tasks with the administrative console or other administrative tools.

After you set up the Network Deployment environment, you mainly need to monitor and control incorporated nodes and the resources on those nodes by using the administrative console or other administrative tools. Use the following tasks to perform these activities.

1. Use the settings page for an administrative service to configure administrative services.
2. Configure cells.
3. Configure deployment managers.
4. Manage nodes.
5. Manage node agents.
6. Manage node groups.
7. Configure remote file services.
8. Configure location service daemons on the z/OS system.

Cells

Cells are logical groupings of one or more nodes in a WebSphere Application Server distributed network.

A cell is a configuration concept, a way for administrators to logically associate nodes with one another. Administrators define the nodes that make up a cell, according to the specific criteria that make sense in their organizational environments.

Administrative configuration data is stored in XML files. A cell retains master configuration files for each server in every node in the cell. Each node and server also have their own local configuration files. Changes to a local node or to a server configuration file are temporary, if the server belongs to the cell. While in effect, local changes override cell configurations. Changes to the master server and master node configuration files made at the cell level replace any temporary changes made at the node when the cell configuration documents are synchronized to the nodes. Synchronization occurs at designated events, such as when a server starts.

Configuring cells

This topic describes how to change the cell protocol information, define custom properties for the cell, and add additional nodes.

Before you can configure cells, you must install the WebSphere Application Server Network Deployment product.

When you create a deployment manager profile, a cell is created. A cell provides a way to group one or more nodes of your Network Deployment product. You probably do not need to configure the cell again. To view information about and to manage a cell, use the settings page for a cell.

1. Access the settings page for a cell. Click **System Administration > Cell** from the navigation tree of the administrative console.
2. If the protocol that the cell uses to retrieve information from a network is not appropriate for your system, select the appropriate protocol. By default, a cell uses Transmission Control Protocol (TCP). If you want the cell to use User Datagram Protocol, select **UDP** from the list for **Cell Discovery Protocol** on the settings page for the cell. It is unlikely that you need to change the cell protocol configuration from TCP.
3. Click **Custom Properties** and define any name-value pairs that your deployment manager needs.

4. When you install the WebSphere Application Server Network Deployment product, a node is added to the cell. You can add additional nodes on the Node page. Click **Nodes** to access the Node page, which you use to manage nodes.

Depending on which steps you performed, you changed the cell protocol information, defined custom properties for the cell, and added additional nodes.

You can continue to administer your Network Deployment product by doing such tasks as managing nodes, node agents, and node groups.

Cell settings

Use this page to set the discovery protocol and address end point for an existing cell. A cell is a configuration concept, a way for an administrator to logically associate nodes according to whatever criteria make sense in the administrator's organizational environment.

To view this administrative console page, click **System Administration > Cell**.

Name

Specifies the name of the existing cell.

A cell name must be unique in any circumstance in which the product is running on the same physical machine or cluster of machines, such as a sysplex. Additionally, a cell name must be unique in any circumstance in which network connectivity between entities is required either between the cells or from a client that must communicate with each of the cells. Cell names also must be unique if their name spaces are going to be federated. Otherwise, you might encounter symptoms such as a javax.naming.NameNotFoundException exception, in which case, you need to create uniquely named cells.

Short Name

Specifies the short name of the cell. The name is 1-8 characters, alphanumeric or national language. It cannot start with a numeric.

The short name property is read only. It was defined during installation and customization.

Cell Discovery Protocol

Specifies the protocol that the cell follows to retrieve information from a network.

Select one of these protocol options:

UDP	User Datagram Protocol (UDP)
TCP	Transmission Control Protocol (TCP)

Default	TCP

Deployment managers

Deployment managers are administrative agents that provide a centralized management view for all nodes in a cell, as well as management of clusters and workload balancing of application servers across one or several nodes in some editions.

WebSphere Application Server for z/OS uses Workload Management (WLM) as the primary vehicle for workload balancing.

A deployment manager hosts the administrative console. A deployment manager provides a single, central point of administrative control for all elements of the entire WebSphere Application Server distributed cell. Each cell contains one deployment manager.

Configuring deployment managers

Configure deployment managers for a single, central point of administrative control for all elements in a WebSphere Application Server distributed cell.

When you created a deployment manager profile, a deployment manager was created. You can run the deployment manager with its default settings. However, you can follow this task to change the deployment manager configuration settings such as the ports the process uses, custom services, logging and tracing settings, and so on. To view information about and manage a deployment manager, use the settings page for a deployment manager.

1. Access the settings page for a deployment manager. Click **System Administration > Deployment Manager** from the navigation tree of the administrative console.
2. Configure the deployment manager as desired by clicking on a property such as **Custom Services** and specifying settings on the resulting pages.
3. If you specify the server short name as 8 characters, follow the directions toconvert the default 7 character short name to 8 characters.

Deployment manager settings

Use this page to stop the deployment manager from running, and to link to other pages which you can use to define additional properties for the deployment manager. A deployment manager provides a single, central point of administrative control for all elements of the entire WebSphere Application Server distributed cell.

To view this administrative console page, click **System administration > Deployment manager**.

Name

Specifies a logical name for the deployment manager. The name must be unique within the cell.

Data type	String

Short name

Specifies the short name of the deployment manager server.

The server short name must be unique within a cell. The short name identifies the server to the native facilities of the operating system, such as Workload Manager (WLM), Automatic Restart Manager, SAF (for example, RACF), started task control, and others.

The name is 1-8 characters, alpha numeric or national language. It cannot start with a numeric.

The system assigns a cell-unique, default short name.

Data type	String

Unique Id

Specifies the unique ID of this deployment manager server.

The unique ID property is read only. The system automatically generates the value.

Data type	String

Process ID

Specifies a string identifying the process.

Data type	String
Default	None

Cell Name

Specifies the name of the cell for the deployment manager. The default is the name of the host computer on which the deployment manager is installed with `Cell##` appended, where *##* is a two-digit number.

Data type	String
Default	*host_name*Cell01

Node Name

Specifies the name of the node for the deployment manager. The default is the name of the host computer on which the deployment manager is installed with `CellManager##` appended, where *##* is a two-digit number.

Data type	String
Default	*host_name*CellManager01

State

Indicates the state of the deployment manager. The state is *Started* when the deployment manager is running and *Stopped* when it is not running.

Data type	String
Default	Started

Node

A *node* is a logical grouping of managed servers.

A node usually corresponds to a logical or physical computer system with a distinct IP host address. Nodes cannot span multiple computers. Node names usually are identical to the host name for the computer.

Nodes in the network deployment topology can be managed or unmanaged. A managed node has a node agent process that manages its configuration and servers. Unmanaged nodes do not have a node agent.

A managed node has a node agent that manages all servers on a node, whether the servers are WebSphere Application Servers, Java Message Service (JMS) servers (on Version 5 nodes only), Web servers, or generic servers. The node agent represents the node in the management cell and keeps the configuration up to date.

An unmanaged node does not have a node agent to manage its servers. Unmanaged nodes in the Network Deployment environment can have server definitions such as Web servers, but not Application Server definitions. Unmanaged nodes in the Network Deployment environment cannot have a node agent added to it, and therefore cannot become a managed node. In the stand-alone Application Server environment, nodes do not have node agents and are also considered unmanaged nodes. The deployment manager cannot manage a stand-alone Application Server because it is not known to the cell. A stand-alone Application Server can be federated. When it is federated, a node agent is automatically created, and the node becomes a managed node in the cell.

A supported Web server can be on a managed node or an unmanaged node. You can define only one Web server to a stand-alone WebSphere Application Server node. This Web server is defined on an unmanaged node. You can define Web servers to the deployment manager. These Web servers can be defined on managed or unmanaged nodes.

WebSphere Application Server supports basic administrative functions for all supported Web servers. For example, the generation of a plug-in configuration can be performed for all Web servers. However, propagation of a plug-in configuration to remote Web servers is supported only for IBM HTTP Servers that are defined on an unmanaged node. If the Web server is defined on a managed node, propagation of the plug-in configuration is done for all the Web servers by using node synchronization. The Web server plug-in configuration file is created according to the Web server definition and is based on the list of applications that are deployed on the Web server. You can also map all supported Web servers as potential targets for the modules during application deployment.

WebSphere Application Server supports some additional administrative console tasks for IBM HTTP Servers on managed and unmanaged nodes. For instance, you can start IBM HTTP Servers, stop them, terminate them, display their log files, and edit their configuration files.

You can add managed and unmanaged nodes to a Network Deployment cell in one of the following ways:

- Administrative console
- Command line (managed nodes only)
- Administrative script
- Java program

Each of these methods for adding a managed node to a Network Deployment cell includes the option of specifying a target node group for the managed node to join. If you do not specify a node group, or you do not have the option of specifying a node group, the default node group of DefaultNodeGroup is the target node group.

On the z/OS system, the default DefaultNodeGroup node group is the sysplex node group for the deployment manager node and any other node in the cell from the same sysplex. A z/OS system node from a different sysplex cannot be a member of this node group and must be a member of a sysplex node group for its sysplex.

Whether you specify an explicit node group or accept the default, the node group membership rules must be satisfied. If the node that you are adding does not satisfy the node group membership rules for the target node group, the add node operation fails with an error message.

Managing nodes

This topic describes how to add a node, select the discovery protocol for a node, define a custom property for a node, stop servers on a node, and remove a node.

A node is a grouping of managed or unmanaged servers. You can add both managed and unmanaged nodes to the WebSphere Application Server topology. If you add a new node for an existing WebSphere Application Server to the Network Deployment cell, you add a managed node. If you create a new node in the topology for managing Web servers or servers other than WebSphere Application Servers, you add an unmanaged node.

To view information about nodes and managed nodes, use the Nodes page. To access the Nodes page, click **System Administration > Nodes** in the administrative console navigation tree.

You can manage nodes on an application server through the wsadmin scripting tool, through the Java application programming interfaces (APIs), or through the administrative console. Perform the following tasks to manage nodes on an application server through the administrative console.

- **Add a node.**
  1. Go to the Nodes page and click **Add Node**. Choose whether you want to add a managed or unmanaged node, and click **Next**.
  2. For a managed node, verify that an application server is running on the remote host for the node that you are adding. On the Add Node page, specify a host name, connector type, and port for the application server at the node you are adding.
  3. For a managed node, perform one of the following sets of actions listed in the table:

If the deployment manager is on	And the node that you add to the cell is on	Complete the appropriate set of actions:
A z/OS system	A z/OS system and is in the same sysplex as the deployment manager	Optionally specify a node group and a core group. Click **OK**.
A z/OS system	A z/OS system, but is on a different sysplex than the deployment manager	Specify a node group that contains nodes from the same sysplex as the node you are adding. If no such node group exists, create a node group and then specify that node group. Optionally specify a core group. Click **OK**.
The distributed platform or the i5/OS platform	A z/OS system	Specify a node group that contains nodes from the same sysplex as the node you are now adding. If no such node group exists, create a node group and then specify that node group. Optionally specify a core group. Click **OK**.
A z/OS system	The distributed platform or the i5/OS platform	Specify a node group that contains distributed nodes. If no such node group exists, create a node group and then specify that node group. Optionally specify a core group. Click **OK**.

     For the node group option to display, a group other than the default node group must first be created. Likewise, for the core group option to display, a group other than the default core group must first be created.
  4. For managed nodes, another administrative console panel is displayed if the node to federate is on a Windows operating system. Specify on the panel whether you want to register the node agent to run as a Windows service. If security is enabled, you can optionally enter the local operating system user name and password under which you will run the service. If you do not specify a user name and password, the service runs under the local system identity. When you run remove the node, the node agent is de-registered as a Window service.
  5. For an unmanaged node, on the **Nodes > New** page, specify a node name, a host name, and a platform for the new node. Click **OK**.

  The node is added to the WebSphere Application Server environment and the name of the node is displayed in the collection on the Nodes page.

  Join subsequent WebSphere Application Server for z/OS nodes from the same sysplex to the same sysplex node group. If you add WebSphere Application Server for z/OS nodes from different sysplexes to the same cell, establish a separate sysplex node group for the nodes of each sysplex.
- **Select the discovery protocol.**

  If the discovery protocol that a node uses is not appropriate for the node, select the appropriate protocol. On the Nodes page, click the node to access the Settings for the node. Select a value for **Discovery Protocol**. User Datagram Protocol (UDP) is faster than Transmission Control Protocol

(TCP). However, TCP is more reliable than UDP because UDP does not guarantee the delivery of datagrams to the destination. The default of TCP is the recommended value.

For a node agent or deployment manager, use **TCP** or **UDP**.

A managed process uses multicast as its discovery protocol. The discovery protocol is fixed for a managed process. The main benefit of using multicast on managed processes is efficiency for the node agent. Suppose you have forty servers in a node. A node agent that uses multicast sends one broadcast to all forty servers. If a node agent did not use multicast, it would send discovery queries to all managed processes one at a time, totaling forty sends. Additional benefits of using multicast are that you do not have to configure the discovery port for each server or prevent port conflicts because all servers in one node listen to one port instead of to one port for each server.

- **Define a custom property for a node.**
  1. On the Nodes page, click the node for which you want to define a custom property.
  2. On the Settings for the node, click **Custom Properties**.
  3. On the Property collection page, click **New**.
  4. On the Settings page for a property instance, specify a name-value pair and a description for the property, and click **OK**.
- Synchronize the node configuration.

  If you add a managed node or change a managed node configuration, synchronize the node configuration. On the Node Agents page, ensure that the node agent for the node is running. Then, on the Nodes page, select the check box beside the node whose configuration files you want to synchronize and click **Synchronize** or **Full Resynchronize**.

  Clicking either option sends a request to the node agent for that node to perform a configuration synchronization immediately, instead of waiting for the periodic synchronization to occur. This action is important if automatic configuration synchronization is disabled, or if the synchronization interval is set to a long time, and a configuration change is made to the cell repository that needs to replicate to that node. Settings for automatic synchronization are on the File Synchronization Service page.

  Synchronize requests that a node synchronization operation be performed using the normal synchronization optimization algorithm. This operation is fast, but might not fix problems from manual file edits that occur on the node. It is still possible for the node and cell configuration to be out of synchronization after this operation is performed.

  Full Resynchronize clears all synchronization optimization settings and performs configuration synchronization anew, so there is no mismatch between node and cell configuration after this operation is performed. This operation can take longer than the **Synchronize** operation.

  Unmanaged nodes cannot be synchronized.
- **Stop servers on a node.**

  On the Nodes page, Select the check box beside the managed node whose servers that you want to stop running, and click **Stop**.
- **Remove a node.**

  On the Nodes page, Select the check box beside the node that you want to delete and click **Remove Node**. If you cannot remove the node by clicking **Remove Node**, remove the node from the configuration by clicking **Force Delete**.
- **View node capabilities.**

  Review the node capabilities, such as the product version through the administrative console. You can also query them through the Application Server application programming interface (API) or the wsadmin tool. For information on the wsadmin tool, see the *Using the administrative clients* PDF.

  The product versions for WebSphere Application Server are as follows: The base edition of WebSphere Application Server is listed in the version column as `Base`. The express edition of WebSphere Application Server is listed in the version column as `Express`. The Network Deployment product is listed in the version column as `ND`.

Node collection

Use this page to manage nodes in the WebSphere Application Server environment. Nodes group managed servers. The table lists the managed and unmanaged nodes in this cell. The first node is the deployment manager. Add new nodes to the cell and to the list by clicking **Add Node**.

To view this administrative console page, click **System administration > Nodes**.

Name

Specifies a name for a node that is unique within the cell.

A node corresponds to a physical computer system with a distinct IP host address. The node name is usually the same as the host name for the computer.

Both Internet Protocol Version 4 (IPv4) and Internet Protocol Version 6 (IPv6) are now supported by WebSphere Application Server, but there are restrictions that apply to using both IPv4 and IPv6 in the same cell. Note that when you add a node to a cell, the format in which you specify the name is based on the version of IP the node will be using.

Version

Specifies the product version of the node.

The product version is the version of a WebSphere Application Server for managed nodes. For unmanaged nodes on which you can define Web servers, the version displays as not applicable

The base edition of WebSphere Application Server is listed in the version column as `Base`. The express edition of WebSphere Application Server is listed in the version column as `Express`. The Network Deployment product is listed in the version column as `ND`.

Discovery protocol

Specifies the protocol that servers use to discover the presence of other servers on this node.

The possible protocol options follow:

UDP User Datagram Protocol (UDP)
TCP Transmission Control Protocol (TCP)

Status

Indicates that the node is either synchronized, not synchronized, unknown, or not applicable.

Status	Explanation
Synchronized	The configuration files on this node are synchronized with the deployment manager.
Not synchronized	The configuration files on this node are not synchronized with the deployment manager and are out-of-date. Perform a synchronize operation to get the latest configuration changes on the node.
Unknown	The state of the configuration file cannot be determined because the node agent cannot be reached for this node.
Not applicable	The status column is not applicable for this node because the node is an unmanaged node.

Node settings

Use this page to view or change the configuration or topology settings for either a managed node instance or an unmanaged node instance.

A managed node is a node with an Application Server and a node agent that belongs to a cell. An unmanaged node is a node defined in the cell topology that does not have a node agent running to manage the process. Unmanaged nodes are typically used to manage Web servers.

To view this administrative console page, click **System administration > Nodes >** *node_name*.

Name:

Specifies a logical name for the node. The name must be unique within the cell.

A node name usually is identical to the host name for the computer. However, you choose the node name. You can make the node name some name other than the host name.

Data type	String

Both Internet Protocol Version 4 (IPv4) and Internet Protocol Version 6 (IPv6) are now supported by WebSphere Application Server, but there are restrictions that apply to using both IPv4 and IPv6 in the same cell. Note that when you add a node to a cell, the format in which you specify the name is based on the version of IP the node will be using.

Short Name:

Specifies the name of a node. The name is 1-8 characters, alphanumeric or national language. It cannot start with a numeric.

The short name property is defined during installation and customization. However, you can change the short name using the **renameNode.sh** command.

Host name:

Specifies the host name of the unmanaged node that is added to the configuration.

Date type	String
Default	None

Discovery Protocol:

Specifies the protocol that the node follows to retrieve information from a network. The Discovery protocol setting is only valid for managed nodes.

Select from one of these protocol options:

UDP	User Datagram Protocol (UDP)
TCP	Transmission Control Protocol (TCP)

Data type	String
Default	TCP
Range	Valid values are UDP or, TCP.

UDP is faster than TCP, but TCP is more reliable than UDP because UDP does not guarantee delivery of datagrams to the destination. Between these two protocols, the default of TCP is recommended.

File permissions: Specifies the most lenient file permissions for the application files that WebSphere Application Server extracts into the application destination location. A deployer can override the permissions by configuring the permissions at the application level. However, if the file permissions specified at the application level are more lenient than the ones specified at the node, the ones specified at the node are used. The File permissions setting is only valid for managed nodes.

Data type	String
Default	755 , or rwx-rx-rx, for files that end in .dll, .so, .a and .sl if no value is set

Platform type:

Specifies the operating system on which the unmanaged node runs.

Valid options are:

Windows
AIX
HP-UX
Solaris
Linux
OS/400
z/OS

Add managed nodes

A managed node is a node with an Application Server and a node agent that belongs to a cell. Use this page to add a managed node to a cell.

To view this administrative console page, click **System Administration > Nodes > Add node > Next** .

Node connection

Specifies connection information for WebSphere Application Server.

- **Host**

  Specifies the host name or IP address of the node to add to the cell. A WebSphere Application Server instance must be running on this machine.

Data type	String
Default	None

- **JMX connector type**

  Specifies the Java Management Extensions (JMX) connectors that communicate with the WebSphere Application Server when you invoke a scripting process.

  Select from one of these JMX connector types:

Simple Object Access Protocol (SOAP)

Use when the Application Server connects to a SOAP server.
Remote Method Invocation (RMI)

Use when the Application Server connects to an RMI server.

- **JMX connector port**

  Specifies the port number of the JMX connector on the instance to add to the cell. The default SOAP connector port is 8880.

Date type	Integer
Default	8880

- **Application server user name**

  Specifies the administration user name that connects to the remote Application Server whose node is being added to the cell. The Application Server user name and password are used to connect to the Application Server and start the add node process at the Application Server. The Application Server user name and password settings always display. You must specify values for them if security is enabled at the Application Server. Otherwise, leave them blank. User name requirements are the requirements that the security system that you use imposes for a user name.
- **Application server password**

  Specifies the password for the Application Server user name that you supply. Password requirements are the requirements that the security system that you use imposes for a password.
- **Deployment manager user name**

  Specifies the deployment manager administration user name that the Application Server uses when connecting to the deployment manager to add its node to the cell. The deployment manager user name and password settings display only if security is enabled at the deployment manager. The deployment manager user name and password are required if their settings display. User name requirements are the requirements that the security system that you use imposes for a user name.
- **Deployment manager password**

  Specifies the password for the deployment manager user name that you supply. Password requirements are the requirements that the security system that you use imposes for a password.

Options

Select from the following settings to further specify characteristics when adding a managed node to a cell.

- **Include Applications**

  Copies the applications installed on the remote instance into a cell. If the applications to copy have the same name as the applications that currently exist in the cell, the Application Server does not copy the applications.
- **Include buses**

  Specifies whether to move the bus configuration at the node to the deployment manager.
- **Starting port**

  Specifies the port numbers for the node agent process.

Use default	Specifies whether to use the default node agent port numbers.
Specify	Allows you to specify the starting port number in the Port number field. WebSphere Application Server administration assigns the port numbers in order from the starting port number. For example, if you specify 9950, the administration program configures the node agent ports as 9950, 9951, 9952, and so on.

- **Core Group**

  Specifies the group to which you can add a cluster or node agent. By default, clusters or node agents are added to the DefaultCoreGroup group.

  Select from one of the core groups if a list is displayed. The list displays if a core group in addition to the default core group exists.
- **Node group**

  Specifies the group to which you can add the node. By default, nodes are added to the DefaultNodeGroup group.

  Select from one of the node groups if a list is displayed. The list displays if a node group in addition to the default node group exists.

Node installation properties

Use this page to view read-only installation properties for this node. These properties provide information about the capabilities of the node that are collected during product installation time, such as the operating system name, architecture and version, or WebSphere Application Server product levels that are installed on the node.

To view this administrative console page, click **System administration > Nodes > *node name* > Node installation properties**.

Information about a node, such as operating system platform and product features, is maintained in the configuration repository in the form of properties. As product features are installed on a node, new property settings are added.

WebSphere Application Server system management uses the managed object metadata properties as follows:

- To display the node version in the administrative console
- To ensure that new configuration types or attributes are not created or set on older release nodes
- To ensure that new resource types are not created on old release nodes
- To ensure that new applications are not installed on old release nodes because the old run time cannot support the new applications

For detailed information about the following properties, see the Application Server application programming interface (API).

com.ibm.websphere.baseProductShortName

The product short name for the WebSphere Application Server that is installed.

com.ibm.websphere.baseProductVersion

The version of WebSphere Application Server that is installed.

com.ibm.websphere.nodeOperatingSystem

The operating system platform on which the node runs.

com.ibm.websphere.nodeSysplexName

The sysplex name on a z/OS operating system.

This property applies to the z/OS operating system only.

Node group

A *node group* is a collection of managed nodes. Managed nodes are WebSphere Application Server nodes. A node group defines a boundary for server cluster formation.

Nodes that you organize into a node group need to be similar in terms of installed software, available resources, and configuration to enable servers on those nodes to host the same applications as part of a server cluster. The deployment manager does no validation to guarantee that nodes in a given node group have anything in common.

Node groups are optional and are established at the discretion of the WebSphere Application Server administrator. However, a node must be a member of a node group. Initially, all Application Server nodes are members of the default DefaultNodeGroup node group.

A node can be a member of more than one node group.

On the z/OS platform, an Application Server node must be a member of a sysplex node group. Nodes in the same sysplex must be in the same sysplex node group. A node can be in one sysplex node group only.

When the deployment manager is configured on a z/OS node, the default node group, DefaultNodeGroup, is the sysplex node group for the deployment manager node and any other node in the cell from the same sysplex. Sysplex node groups are special node groups that the system manages.

Nodes on distributed platforms and i5/OS platforms cannot be members of a node group that contains a node on a z/OS platform. However, nodes on distributed platforms and nodes on i5/OS platforms can be members of the same node group.

To delete a node group, the node group must be empty. The default node group cannot be deleted.

Node group membership rules

Nodes can be members of node groups if they meet certain requirements.

Node group membership must adhere to the following rules:

- A node in a node group must be a managed node.
- A managed node must be a member of at least one node group.
- If the node is on the z/OS platform, the node must be a member of a sysplex node group. The node can also be a member of other node groups that are not sysplex node groups.
- Nodes on distributed platforms and i5/OS platforms cannot be members of a node group that contains a node on a z/OS platform.
- Nodes on the z/OS platform that are in different sysplexes must be members of different node groups.

Sysplex node groups

A sysplex node group is a node group unique to the z/OS operating system. The sysplex node group includes a sysplex name and a z/OS operating system location service configuration. A sysplex is a collection of z/OS systems that cooperate by using certain hardware and software products to process workloads.

You cannot explicitly create a sysplex node group. The z/OS operating system creates sysplex node groups in the following ways:

- When you configure a deployment manager server on the z/OS operating system, the default node group is a sysplex node group. The deployment manager is automatically a member of the sysplex node group. Application Server for z/OS nodes that you add to the network deployment cell are automatically members of this node group.
- You can add an Application Server for z/OS node to a network deployment cell whose deployment manager is on a distributed platform node. In this case, you must add the first Application Server for z/OS node for the network deployment cell to an empty node group. The system automatically configures the node group into a sysplex node group by using the sysplex name and the z/OS location service configuration that belongs to the Application Server for z/OS node.

You cannot remove a node from a sysplex node group. However, if a node is the only member of a sysplex node group, you can add that node to an empty node group. The empty node group is converted into a sysplex node group and the former sysplex node group of the node is converted into a regular node group.

You cannot delete a node group that is a sysplex node group.

Examples: Using node groups

Use node groups to define groups of nodes that are capable of hosting members of the same cluster. An application that is deployed to a cluster must be capable of running on any of the cluster members. The node that hosts each of the cluster members must be configured with software and settings that are necessary to support the application.

By organizing nodes that satisfy your application requirements into a node group, you establish an administrative policy that governs which nodes can be used together to form a cluster. The people who define the cell configuration and the people who create server clusters can operate with greater independence from one another, if they are different people.

Example 1

Assume the following information:

- A cell is comprised of nodes one to eight.
- Each node is a managed node, which means that each node is configured with an Application Server.
- Nodes six, seven, and eight are additionally configured as WebSphere Business Integration Server Foundation nodes.
- All nodes are either z/OS system nodes from the same sysplex, or distributed platform nodes.
- By default, all the nodes are in the default DefaultNodeGroup node group.

Applications that exploit WebSphere Business Integration Server Foundation functions can run successfully only on nodes six, seven, and eight. Therefore, clusters that host these applications can be formed only on nodes six, seven, and eight. To define a clustering policy that guides users of your WebSphere cell into building clusters that can span only predetermined nodes, create an additional node group called WBINodeGroup, for example. Add to the node group nodes six, seven, and eight. If you create a cluster on a node from the WBINodeGroup node group, the system allows only nodes from the WBINodeGroup node group to be members of the cluster.

Example 2

Assume the following information:

- A cell is comprised of nodes one to six.
- Each node is a managed node, which means that each node is configured with an Application Server.
- Nodes one to four are on distributed platforms.
- Nodes five and six are nodes on the z/OS operating system and are in the PLEX1 sysplex.
- The deployment manager is on a distributed platform node.
- Nodes one to four are members of the DefaultNodeGroup node group by default.
- You created empty PLEX1NodeGroup node group to group the z/OS operating system nodes on the PLEX1 sysplex.
- You joined the nodes on the z/OS operating system to the PLEX1NodeGroup node group when you added them to the cell. Nodes on the z/OS operating system cannot be in the same node group with the distributed platform nodes.

Applications that exploit z/OS functions in the PLEX1 sysplex can run successfully on nodes five and six only. Therefore, clusters that host these applications can be formed only on nodes five and six. The required separation of distributed platform nodes from z/OS system nodes establishes a natural clustering policy that guides users of your Application Server cell into building clusters that can span only predetermined nodes. If you create a cluster on a node from the PLEX1NodeGroup node group, the system allows only nodes from the PLEX1NodeGroup node group to be members of the cluster.

Managing node groups

This task discusses how to create and manage node groups.

Read about Nodes groups if you are unfamiliar with them.

Your WebSphere Application Server environment has a default node group. However, if you need additional node groups to manage your Application Server environment, you can create and configure additional node groups. You can delete a node group as long as it is not a default node group.

- View and configure node groups.
  1. Click **System Administration > Node groups** in the console navigation tree.
  2. To view additional information about a particular node group or to further configure a node group, click on the node group name under **Name**.
- Create a node group.
  1. Click **System Administration > Node groups** in the console navigation tree.
  2. Click **New**.
  3. Specify the node group name and description.

  The node group is added to the WebSphere Application Server environment . The name of the node group appears in the name column of the Node group page.

  You can now add nodes to the node group.
- Delete a node group if the node group is not the default node group.
  1. If the node group contains members, delete the members:
     a. Click **System Administration > Node groups** in the console navigation tree.
     b. Under **Name**, click the node group whose members you want to delete.
     c. Click **Node group members**.
     d. Select all the node group members.
     e. Click **Remove**.
  2. Click **System Administration > Node groups**.
  3. Select an empty node group.
  4. Click delete.

Node group collection

Use this page to manage node groups. A node group is a collection of WebSphere Application Server nodes. A node group defines a boundary for server cluster formation.

Nodes that are organized into a node group should be enough alike in terms of installed software, available resources, and configuration to enable servers on those nodes to host the same applications as part of a server cluster. The deployment manager does no validation to guarantee that nodes in a given node group have anything in common.

Node groups are optional and are established at the discretion of the WebSphere administrator. However, a node must be a member of a node group. Initially, all Application Server nodes are members of the default node group. The default node group is DefaultNodeGroup.

A node can be a member of more than one node group.

On the z/OS platform, an Application Server node must be a member of a sysplex node group. Nodes in the same sysplex must be in the same sysplex node group. A node can only be in one sysplex node group. Sysplex node groups are special node groups that the system manages.

A node on a distributed platform and a node on a z/OS platform cannot be members of the same node group.

To delete a node group, the node group must be empty. The default node group cannot be deleted.

To view this administrative console page, click **System Administration > Node groups**.

Name

Specifies a name for a node group that is unique within the cell.

Members

Specifies the number of members or nodes in the node group.

Description

Specifies a description that you define for the node group.

Node group settings

Use this page to view or change the configuration or topology settings for a node group instance.

To view this administrative console page, click **System Administration > Node groups >** *node group name*.

Name:

Specifies a logical name for the node group. The name must be unique within the cell. The name can start with a number.

Data type	String
Maximum length	64 characters

Short name:

Specifies the name of a node. The name must contain 1-8 characters, which are either alphanumeric or national language. It cannot start with a number.

On the z/OS system the short name property is:

- Read-only
- Used only by sysplex node groups
- Defined during installation and customization

Sysplex:

Specifies the name of a node. The name is eight characters, alphanumeric or national language. It cannot start with a numeric. It is used only by sysplex node groups on the z/OS platform. It is defined during installation and customization on z/OS platforms only.

The Sysplex property is read only.

Members:

Specifies the number of nodes within the node group.

Data type	Integer

Description:

Specifies the description that you define for the node group. The description has no specific maximum length.

Managing node group members

Use this topic to manage the nodes in your node groups by viewing, adding or deleting the nodes in a node group.

Read about Nodes groups and Node group membership rules if you are unfamiliar with them.

All nodes must be a member of at least one node group. Initially, all Application Server nodes are members of the default node group named DefaultNodeGroup. Make the nodes that you organize into a node group enough alike in terms of installed software, available resources, and configuration to enable servers on those nodes to host the same applications as part of a server cluster.

- View node groups members.
  1. Click **System Administration** > **Node groups** > ***node group name*** > **Nodes** > **Node group members** in the console navigation tree.
  2. To view additional information about a particular node group member for this node group, click on the node group member name under **Name**.
- Add a node to a node group.
  1. Click **System Administration** > **Node groups** > ***node group name*** > **Nodes** > **Node group members** in the console navigation tree.
  2. Click **Add**.
  3. Select the node from a list. The node group member name is the node name.

  The node group member is added to the node group specified on the breadcrumb trail. The name of the node group member appears in the name column of the Node group member page. You can add additional nodes of similar characteristics to the node group by repeating the steps for adding a node to a node group.

  If the node you add does not satisfy the node group membership rules for the target node group, the add node operation fails with an error message.
- Remove a node from a node group.
  1. Click **System Administration** > **Node groups** > ***node group name*** > **Nodes** > **Node group members** in the console navigation tree.
  2. Select the box next to each node group member that you want to remove from the node group.
  3. Click **Remove**.

  Each node group member that you selected is removed from the node group specified on the breadcrumb trail.

Node group member collection

Use this page to manage node groups members. A node group member is a WebSphere Application Server node.

Click **Add** to add node members to the node group. Click **Remove** to remove node members from the node group.

To view this administrative console page, click **System Administration** > **Node groups** > *node group name* > **Node group members**.

Name

Specifies the name of a node group member.

Node group member settings

Use this page to view or change the configuration or topology settings for a node group member.

To view this administrative console page, click **System Administration** > **Node groups** > *node group name* > **Node group members** > *node group member name*.

Name:

Specifies a logical name for the node group member. A node group member is a node. The name must be unique within the cell.

A node group member name usually is identical to the host name for the computer.

Data type	String
Maximum length	64 characters

The name must contain alphanumeric or national language characters and can start with a number.

Node agents

Node agents are administrative agents that route administrative requests to servers.

A node agent is a server that runs on every host computer system that participates in the WebSphere Application Server Network Deployment product. It is purely an administrative agent and is not involved in application serving functions. A node agent also hosts other important administrative functions such as file transfer services, configuration synchronization, and performance monitoring.

Managing node agents

Node agents are administrative agents that represent a node to your system and manage the servers on that node. Node agents monitor application servers on a host system and route administrative requests to servers. A node agent is created automatically when a node is added to a cell. This topic describes how to view information about a node agent, stop and start the processing of a node agent, and stop and restart application servers on the node that is managed by the node agent.

Before you can manage a node agent, you must install the Network Deployment product.

You can manage nodes through the wsadmin scripting tool, through the Java application programming interfaces (APIs), or through the administrative console. Perform the following tasks to manage nodes on an application server through the administrative console.

- View information about a node agent. Use the Node Agents page. Click **System Administration > Node Agents** in the console navigation tree. To view additional information about a particular node agent or to further configure a node agent, click the node agent name under **Name**.
- Stop and then restart all application servers on the node that is managed by the node agent. On the Node Agents page, select the check box beside the node agent that manages the node whose servers you want to restart, then click **Restart all Servers on Node**.

  Note that the node agent for the node must be processing in order to restart application servers on the node.
- Stop the processing of a node agent. On the Node Agents page, select the check box beside the node agent that you want to stop processing; then click **Stop**.

Depending on the steps that you completed, you have viewed information about a node agent, stopped and started the processing of a node agent, and stopped and restarted application servers on the node that is managed by the node agent.

You can administer other aspects of the Network Deployment environment, such as the deployment manager, nodes, and cells.

Node agent collection

Use this page to view information about node agents. Node agents are administrative agents that monitor application servers on a host system and route administrative requests to servers. A node agent is the running server that represents a node in a Network Deployment environment.

To view this administrative console page, click **System Administration > Node Agents** .

Name

Specifies a logical name for the node agent server.

Node

Specifies a name for the node. The node name is unique within the cell.

A node name usually is identical to the host name for the computer. That is, a node usually corresponds to a physical computer system with a distinct IP host address.

However, the node name is a purely logical name for a group of servers. You can name the node anything you please. The node name does not have to be the host name.

Both Internet Protocol Version 4 (IPv4) and Internet Protocol Version 6 (IPv6) are now supported by WebSphere Application Server, but there are restrictions that apply to using both IPv4 and IPv6 in the same cell. Note that when a node is added to a cell, the format in which the name is specified is based on the version of IP the node will be using.

Version

Specifies the product version of the node.

The product version is the version of a WebSphere Application Server node agent and Application Servers that run on the node.

Status

Indicates whether the node agent server is started or stopped.

Note that if the status of servers such application servers is *Unavailable*, the node agent is not running in the servers' node and you must restart the node agent before you can start the servers.

Node agent server settings

Use this page to view information about and configure a node agent. A node agent coordinates administrative requests and event notifications among servers on a machine. A node agent is the running server that represents a node in a Network Deployment environment.

To view this administrative console page, click **System Administraton > Node Agents >** *node_agent_name*.

A node agent must be started on each node in order for the deployment manager node to be able to collect and control servers configured on that node. If you use configuration synchronization support, a node agent coordinates with the deployment manager server to synchronize the node's configuration data with the master copy managed by the deployment manager.

You must initially start a node agent outside the administrative console. For information on how to initially start a node agent, see the WebSphere Application Server Information Center.

The Runtime tab displays only when a node agent runs.

Name:

Specifies a logical name for the node agent server.

Data type	String

Node Name:

Specifies the name of the node for the node agent server.

Both Internet Protocol Version 4 (IPv4) and Internet Protocol Version 6 (IPv6) are now supported by WebSphere Application Server, but there are restrictions that apply to using both IPv4 and IPv6 in the same cell. Note that when a node is added to a cell, the format in which the name is specified is based on the version of IP the node will be using.

Data type	String

Short name:

Specifies the short name of the node agent server.

The server short name must be unique within a cell. The short name identifies the server to the native facilities of the operating system, such as Workload Manager (WLM), Automatic Restart Manager, SAF (for example, RACF), started task control, and others.

The name is 1-8 characters, alpha-numeric or national language. It cannot start with a numeric.

The system assigns a cell-unique, default short name.

Unique Id:

Specifies the unique ID of this node agent server.

The unique ID property is read only. The system automatically generates the value.

Process ID:

Specifies a string identifying the process.

Data type	String

Cell Name:

Specifies the name of the cell for the node agent server.

Data type	String
Default	*host_name*Network

Node Name:

Specifies the name of the node for the node agent server.

Both Internet Protocol Version 4 (IPv4) and Internet Protocol Version 6 (IPv6) are now supported by WebSphere Application Server, but there are restrictions that apply to using both IPv4 and IPv6 in the same cell. Note that when a node is added to a cell, the format in which the name is specified is based on the version of IP the node will be using.

Data type	String

State:

Indicates whether the node agent server is started or stopped.

Data type	String
Default	Started

Administration service settings

Use this page to view and change the configuration for an administration service.

To view this administrative console page, click **Servers > Application Servers >** *server_name* **> Administration > Administration Services**

Preferred Connector

Specifies the preferred JMX Connector type. Available options, such as SOAPConnector or RMIConnector, are defined using the JMX Connectors page.

Data type	String
Default	SOAP

Extension MBean Providers collection

Use this page to view and change the configuration for JMX extension MBean providers.

You can configure JMX extension MBean providers to be used to extend the existing WebSphere managed resources in the core administrative system. Each MBean provider is a library containing an implementation of a JMX MBean and its MBean XML Descriptor file.

To view this administrative console page, click **Servers > Application Servers >** *server_name* **> Administration > Administration Services > Extension MBean Providers**

Name The name used to identify the Extension MBean provider library.

Description
An arbitrary descriptive text for the Extension MBean Provider configuration.

Classpath
The path to the Java archive (JAR) file that contains the Extension MBean provider library. This class path is automatically added to the Application Server class path.

Extension MBean Provider settings

Use this page to view and change the configuration for a JMX extension MBean provider.

You can configure a library containing an implementation of a JMX MBean, and its MBean XML Descriptor file, to be used to extend the existing WebSphere managed resources in the core administrative system

To view this administrative console page, click **Servers > Application Servers >** *server_name* **> Administration > Administration Services > Extension MBean Providers >** *provider_library_name*

Classpath:

The path to the Java archive (JAR) file that contains the Extension MBean provider library. This class path is automatically added to the Application Server class path. The class loader needs this information to load and parse the Extension MBean XML Descriptor file.

Data type	String

Description:

An arbitrary descriptive text for the Extension MBean Provider configuration. Use this field for any text that helps identify or differentiate the provider configuration.

Data type	String

Name:

The name used to identify the Extension MBean provider library.

Data type	String

Extension MBean collection

You can configure Java Management Extension (JMX) MBeans to extend the existing WebSphere Application Server managed resources in the administrative console. Use this page to register JMX MBeans. Any MBeans that are listed have already been registered.

To view this administrative console page, click **Servers > Application Servers >** *server name* **> Administration > Administration Services > Extension MBean Providers >** *provider library name***> extensionMBeans**

DescriptorURI
Specifies the location, relative to the provider class path, where the MBean XML descriptor file is located.

Type Specifies the type to use for registering this MBean. The type must match the type that is declared in the MBean descriptor file.

Extension MBean settings

Use this page to view and configure Java Management Extension (JMX) MBeans.

To view this administrative console page, click **Servers > Application Servers >** *server name* **> Administration > Administration Services > Extension MBean Providers >** *provider library name* **> ExtensionMBeans >** *descriptorURI*

descriptorURI:

Specifies the location, relative to the provider class path, where the MBean XML descriptor file is located.

Data type	String

type:

Specifies the type to use for registering this MBean. The type must match the type that is declared in the MBean descriptor file.

Data type	String

Java Management Extensions connector properties

You can specify or set a property in the administrative console, the wsadmin tool, Application Server commands, the scripts that run from a command-line interface, or a custom Java administrative client program that you write. You can also set SOAP connector properties in the `soap.client.props` file.

A Java Management Extensions (JMX) connector can either be a Remote Method Invocation (RMI) connector or a Simple Object Access Protocol (SOAP) connector.

For specific information on how to code the JMX connector properties for the wsadmin tool, the Application Server commands, or scripts, see the particular tool or command. For specific information on how to code the JMX connector properties for a custom Java administrative client program, see the Java API documentation for Application Server.

For the administrative console, this topic specifies the coding of the particular setting or property. Coding of properties in the `soap.client.props` file that are specific to JMX connectors is specified. These properties begin with com.ibm.SOAP. Other properties in the `soap.client.props` file that contain information that can be set elsewhere in the Application Server are not documented here. The coding for the com.ibm.ssl.contextProvider property, which can be set only in the `soap.client.props` file, is specified.

Each profile has a property file at `*installation root*/profiles/*profile name*/properties/soap.client.props`. These property files allows you to set different properties, including security and timeout properties. These properties are the default for all the administrative connections that use the SOAP JMX connector between processes that run in a particular profile. For instance, the wsadmin program running under a particular profile uses the property values from that file for the SOAP connector behavior unless the properties are overridden by some other programmatic means.

To view the JMX connector custom properties administrative console panel that goes with this article, click **Servers > Application servers >***server name* **> Server Infrastructure > Administration > Administration Services > Additional properties > JMX Connectors>***connector type* **> Additional Properties > Custom properties**.

SOAP connector properties

This section discusses JMX connector properties that pertain to SOAP connectors.

SOAP request timeout

The value that you choose depends on a number of factors, such as the size and the number of the applications that are installed on the server, the speed of your machine, and the usage of your machine.

The program default value for the request timeout is `600 seconds`. However, other components that connect to the SOAP client can override the default. Components that use the `soap.client.props` file have a default value of `180 seconds`.

Set the property by using one of the following options:

- Scripts that run from a command-line interface.
- The `soap.client.props` file.

Property	com.ibm.SOAP.requestTimeout
Data type	Integer
Range in seconds	0 to n If the property is zero (0), the request never times out.
Default	180

- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	requestTimeout
Data type	Integer
Range in seconds	0 to n If the property is zero (0), the request never times out.
Default	600

- A Java administrative client. The property is AdminClient.CONNECTOR_SOAP_REQUEST_TIMEOUT.

Configuration URL

Specify the configuration Universal Resource Locator (URL) property if you want a program to read SOAP properties from this file. You can set the property by using one of the following options:

- Scripts run from a command-line interface. Scripts can pass the Configuration URL property to the Application Server on the com.ibm.SOAP.ConfigURL system property.
- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	ConfigURL
Data type	String
Valid Value	http://*Path*/soap.client.props
Default	None

- A Java administrative client. Use the AdminClient.CONNECTOR_SOAP_CONFIG property.

Security context provider

This property indicates the Secure Sockets Layer (SSL) implementation to use between the Application Server and the SOAP client. You can specify either IBM Java Secure Sockets Extension (IBMJSSE) or IBM Java Secure Sockets Extension that has undergone Federal Information Processing Standards certification (IBMJSSEFIPS). For information about IBMJSSEFIPS, see the *Using the administrative clients* PDF.

Set the property by using the `soap.client.props` file.

Property	com.ibm.ssl.contextProvider
Data type	String

Valid Values	`IBMJSSE`
	`IBMJSSEFIPS`
	`IBMJSSE2`

Default	IBMJSSE2

Secure Sockets Layer (SSL) security

Use this property to enable SSL security between Application Server and the SOAP client. Set the property by using one of the following options:

- Scripts that run from a command-`line interface.
- The `soap.client.props` file.

Property	com.ibm.SOAP.securityEnabled
Data type	Boolean
Default	False

- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	securityEnabled
Data type	Boolean
Default	False

- A Java administrative client. Use the AdminClient.CONNECTOR_SECURITY_ENABLED property.

SOAP and RMI connector properties

This section discusses JMX connector properties that pertain to both SOAP connectors and RMI connectors.

Connector type

A connector type of SOAP or RMI, depends on whether Application Server connects to a SOAP server or an RMI server. You can set the property by using one of the following options:

- The wsadmin tool.
- Scripts that run from a command-line interface.
- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	Type
Data type	String
Valid values	`SOAPConnector` `RMIConnector`
Default	SOAPConnector

- A Java administrative client. Use the AdminClient.CONNECTOR_TYPE property. Specify the connector type by using the AdminClient.CONNECTOR_TYPE_RMI or the AdminClient.CONNECTOR_TYPE_SOAP constants.

Host

The host name or the IP address of the server to which Application Server connects. The server can be a SOAP server or an RMI server. You can set the property by using one of the following options:

- The wsadmin tool.
- Scripts that run from a command-line interface.
- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	host
Data type	String
Valid values	Host name or IP address
Default	None

- A Java administrative client. Use the AdminClient.CONNECTOR_HOST property.

Port

The port number of the server to which Application Server connects. The server can be a SOAP server or an RMI server. You can set the property by using one of the following options:

- The wsadmin tool.
- Scripts run from a command-line interface.

- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	port
Data type	Integer
Valid value	Port number
Default	None

- A Java administrative client. Use the AdminClient.CONNECTOR_PORT property.

User name

The user name that Application Server uses to access the SOAP server or the RMI server. You can set the property by using one of the following options:

- The wsadmin tool.
- Scripts run from a command-line interface.
- The `soap.client.props` file.

Property	com.ibm.SOAP.loginUserid
Data type	String
Valid value	The value must match the global SSL settings for SOAP or RMI.
Default	None

- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	username
Data type	String
Valid value	The value must match the global SSL settings for SOAP or RMI.
Default	None

- A Java administrative client. Use the AdminClient.USERNAME property.

Password

The password that Application Server uses to access the SOAP server or the RMI server. You can set the property by using one of the following options:

- The wsadmin tool.
- Scripts run from a command-line interface.
- The `soap.client.props` file.

Property	com.ibm.SOAP.loginPassword
Data type	String
Valid values	The value must match the global SSL settings for SOAP or RMI.
Default	None

- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	password
Data type	String

Valid values	The value must match the global SSL settings for SOAP or RMI.
Default	None

- A Java administrative client. Use the AdminClient.PASSWORD property.

RMI connector properties

This section discusses JMX connector properties that pertain to RMI connectors.

Disabling the JSR 160 RMI connector

Support for JMX Remote application programming interface (JSR 160) is enabled by default so that you automatically receive specification-compliant JMX function. To disable the function for a particular server, set the property by using one of the following options:

- The wsadmin tool.
- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	disableJDKJMXConnector
Data type	string
Value	true

Java Management Extensions connectors

Use this page to view and change the configuration for Java Management Extensions (JMX) connectors.

To view this administrative console page, click one of the following paths:

- **Servers** > **Application Servers** > *server_name* > **Administration** > **Administration Services** > **JMX Connectors**
- **Servers** > **JMS Servers** > *server_name* > **Administration** > **Administration Services** > **JMX Connectors**

Java Management Extensions (JMX) connectors communicate with WebSphere Application Server when you invoke a scripting process. There is no default for the type and parameters of a connector. The `wsadmin.properties` file specifies the Simple Object Access Protocol (SOAP) connector and an appropriate port number. You can also use the Remote Method Invocation (RMI) connector.

Use one of the following methods to select the connector type and attributes:

- Specify properties in a properties file.
- Indicate options on the command line.

On the z/OS platform, both the SOAP connector and the RMI connector connect to a controller (server) . WebSphere Application Server for z/OS internally routes MBean requests from a controller to its servant regions as appropriate.

Type

Specifies the type of the JMX connector.

Data type	Enum
Default	SOAPConnector

Range	**SOAPConnector** For JMX connections using Simple Object Access Protocol (SOAP). **RMIConnector** For JMX connections using Remote Method Invocation (RMI).

JMX connector settings

Use this page to view the configuration for a Java Management Extensions (JMX) connector.

To view this administrative console page, click one of the following paths:

- **Servers** > **Application Servers** > *server_name* > **Administration** > **Administration Services** > **JMX Connectors** > *connector_type*
- **Servers** > **JMS Servers** > *server_name* > **Administration** > **Administration Services** > **JMX Connectors** > *connector_type*

Both the SOAP connector and the RMI connector connect to a controller (server) . WebSphere Application Server for z/OS internally routes MBean requests from a controller to its servant regions as appropriate.

Type:

Specifies the type of the JMX connector.

Data type	Enum
Default	SOAPConnector
Range	**SOAPConnector** For JMX connections using Simple Object Access Protocol (SOAP). **RMIConnector** For JMX connections using Remote Method Invocation (RMI).

Repository service settings

Use this page to view and change the configuration for an administrative service repository.

To view this administrative console page, click **Servers** > **Application Servers** > *server_name* **Administration** > **Administration Services** > **Repository Service**.

Audit Enabled

Specifies whether to audit repository updates in the log file. The default is to audit repository updates.

Data type	Boolean
Default	true

Administration services custom properties

This topic discusses the administration services custom properties that you can set on the administrative console.

To view the administration services custom properties administrative console page that goes with this topic, click: **Servers** > **Application Server** > *server_name* > **Administration** > **Administration Services** > **Custom Property**.

Specify a property and its value as a name-value pair on the Administration services custom properties page.

Disable routing

When a custom managed bean (MBean) is registered directly with the MBean server that runs in a WebSphere Application Server process, the MBean object name is enhanced by default to include the cell, node, and process names as key properties.

With this enhancement, in a Network Deployment environment, the MBean that is registered on an application server is addressable through a client that is connected to the deployment manager.

To turn off the default behavior, set the following custom property on the application server:

Property name
com.ibm.websphere.mbeans.disableRouting

Data type
string

Value
One or more MBean object names tagged with `<on>...</on>`. You can specify the object name of your MBean or a pattern that matches the names of several MBeans.

Example:

If you register a custom MBean with the `WebSphere:type=custom,name=custommbean1` object name and another custom MBean with the `WebSphere:type=custom,name=custommbean2` object name, each of the following values is valid:

- `<on>WebSphere:type=custom,name=custommbean1</on>`

  The value disables the MBean object name modification for this MBean.
- `<on>WebSphere:type=custom,*</on>`

  The value disables the MBean object name modification for this MBean.
- `<on>WebSphere:type=custom,name=custommbean1</on><on>WebSphere:type=custom,name=custommbean2</on>`

  The value disables the object name modification for both MBeans.

If this custom property is set, an administrative client needs to connect directly to the application server on which the MBean is registered to invoke methods. The MBean cannot participate in all the distributed functions of the administrative system.

Administrative audits

This topic discusses aspects of administrative audits, such as log files that contain the audit information, the administrative actions that are audited, and the types of audit messages that are logged.

Administrative audits use the same logging facility as the rest of the product. The audits are available in both the `activity.log` file and the `SystemOut.log` of the server that performs the action. You do not need to enable trace to produce the audits. However, through the Repository service console page, you can control whether configuration change auditing is done. This type of audit is done by default. Operational command auditing is always enabled. Information about which user performed the change is available only when security is enabled.

The following administrative actions are audited:

- All configuration changes, in terms of the configuration documents that are created, modified, or deleted.
- Certain operational changes like starting and stopping nodes, clusters, servers, and applications. These managed bean (MBean) operations provide administrative auditing:

Table 2.

MBean type	**MBean operations**
CellSync	syncNode
Cluster	start, stop, stopImmediate, rippleStart
NodeAgent	launchProcess, stopNode, restart
Server	stop, stopImmediate
AppManagement	startApplication, stopApplication

Configuration change audits have ADMRxxxxI message IDs, where xxxx is the message number. Operational audits have ADMN10xxI message IDs, where 10xx is the message number.

Here are some examples from the deployment manager `SystemOut.log` file:

```
[7/23/03 17:04:49:089 CDT] 39c26dad FileRepositor A ADMR0015I: Document
cells/ellingtonNetwork/security.xml was modified by user u1.
   [7/23/03 17:04:49:269 CDT] 3ea0edb5 FileRepositor A ADMR0016I: Document
cells/ellingtonNetwork/nodes/ellington/app.policy was created by user u1.
   ...
   [7/23/03 17:13:54:081 CDT] 39a572a1 AdminHelper   A ADMN1008I: Attempt
made to start the SamplesGallery application. (User ID = u1)
   ...

and from the node agent SystemOut.log...
[7/23/03 17:38:43:461 CDT]  23d1326 AdminHelper   A ADMN1000I: Attempt
made to launch server1 on node ellington. (User ID = u1)

and from the app server SystemOut.log...
[7/23/03 17:39:59:360 CDT] 24865373 AdminHelper   A ADMN1020I: Attempt
made to stop the server1 server. (User ID = u1)
```

The message text is split for printing purposes.

Remote file services

Configuration documents describe the available application servers, their configurations, and their contents. Two file services manage configuration documents: the file transfer service and the file synchronization service.

The following information describes what the file services do:

File transfer service

The file transfer service enables the moving of files between the network manager and the nodes. It uses the HTTP protocol to transfer files. When you enable security in the WebSphere Application Server product, the file transfer service uses certificate-based mutual authentication. You can use the default key files in a test environment. Ensure that you change the default key file to secure your system.

The ports used for file transfer are the HTTP_Transport port, the HTTPS transport port, the administrative console port, and the administrative console secure port. For more information, see the Planning for TCP/IP port convention topic in the *Installing your application serving environment* PDF.

File synchronization service

The file synchronization service ensures that a file set on each node matches that on the deployment manager node. This service promotes consistent configuration data across a cell. You can adjust several configuration settings to control file synchronization on individual nodes and throughout a system.

This service runs in the deployment manager and node agents, and ensures that configuration changes made to the cell repository are propagated to the appropriate node repositories. The cell repository is the master repository, and configuration changes made to node repositories are not propagated up to the cell. During a synchronization operation a node agent checks with the deployment manager to see if any configuration documents that apply to the node have been updated. New or updated documents are copied to the node repository, and deleted documents are removed from the node repository.

The default behavior, which is enabled, is for each node agent to periodically run a synchronization operation. You can configure the interval between operations or disable the periodic behavior. You can also configure the synchronization service to synchronize a node repository before starting a server on the node.

Configuring remote file services

Configuration data for the WebSphere Application Server product resides in files. Two services help you reconfigure and otherwise manage these files: the file transfer service and file synchronization service.

By default, the file transfer service is always configured and enabled at a node agent, so you do not need to take additional steps to configure this service. However, you might need to configure the file synchronization service.

1. Go to the File Synchronization Service page. Click **System Administration > Node Agents** in the console navigation tree. Then, click the node agent for which you want to configure a synchronization server and click **File Synchronization Service**.
2. On the File Synchronization Service page, customize the service that helps make configuration data consistent across a cell by moving updated configuration files from the deployment manager to the node. Change the values for properties on the File Synchronization Service page. The file synchronization service is always started, but you can control how it runs by changing the values.

File transfer service settings

Use this page to configure the service that transfers files from the deployment manager to individual remote nodes.

To view this administrative console page, click **System Administration > Node Agents >** *node_agent_name* **> File Transfer Service**.

Enable service at server startup

Specifies whether the server attempts to start the specified service. Some services are always enabled and disregard this property if set. This setting is enabled by default.

Data type	Boolean
Default	true

Retries count

Specifies the number of times you want the file transfer service to retry sending or receiving a file after a communication failure occurs.

Data type	Integer

Default	3 If the retries count setting is blank, the file transfer service sets the default to 3. If the retries count setting is 0, the file transfer service does not retry. The default is the recommended value.

Retry wait time

Specifies the number of seconds that the file transfer service waits before it retries a failed file transfer.

Data type	Integer
Default	10 If the retry wait time setting is blank, the code sets the default to 10. If the retry wait time setting is 0, the file transfer service does not wait between retries. The default is the recommended value.

File synchronization service settings

Use this page to specify that a file set on one node matches that on the central deployment manager node and to ensure consistent configuration data across a cell.

You can synchronize files on individual nodes or throughout your system.

To view this administrative console page, click **System Administration** > **Node Agents** > *node_agent_name* > **File Synchronization Service**.

Enable service at server startup

Specifies whether the server attempts to start the file synchronization service. This setting does not cause a file synchronization operation to start. This setting is enabled by default.

Data type	Boolean
Default	true

Synchronization Interval

Specifies the number of minutes that elapse between synchronizations. Increase the time interval to synchronize files less often. Decrease the time interval to synchronize files more often.

Data type	Integer
Units	Minutes
Default	10 The minimum value that the application server uses is 1. If you specify a value of 0, the application server ignores the value and uses the default of 1.

Automatic Synchronization

Specifies whether to synchronize files automatically after a designated interval. When this setting is enabled, the node agent automatically contacts the deployment manager every synchronization interval to attempt to synchronize the node's configuration repository with the master repository owned by the deployment manager.

If the Automatic synchronization setting is enabled, the node agent attempts file synchronization when it establishes contact with the deployment manager. The node agent waits the synchronization interval before it attempts the next synchronization.

For the z/OS platform, disablement of automatic synchronization is recommended when in a production environment or if you use the application rollout update capability.

Remove the check mark from the check box if you want to control when files are sent to the node.

Data type	Boolean
Default	true

Startup Synchronization

Specifies whether the node agent attempts to synchronize the node configuration with the latest configurations in the master repository prior to starting an application server.

The default is to not synchronize files prior to starting an application server. Enabling the setting ensures that the node agent has the latest configuration but increases the amount of time it takes to start the application server.

Note that this setting has no effect on the startServer command. The startServer command launches a server directly and does not use the node agent.

Data type	Boolean
Default	false

Exclusions

Specifies files or patterns that should not be part of the synchronization of configuration data. Files in this list are not copied from the master configuration repository to the node, and are not deleted from the repository at the node.

The default is to have no files specified.

To specify a file, use a complete name or a name with a leading or trailing asterisk (*) for a wildcard. For example:

`cells/`*`cell name`*`/nodes/`*`node name`*`/`*`file name`*	Excludes this specific file
`*/`*`file name`*	Excludes files named *file name* in any context
`dirname/*`	Excludes the subtree under `dirname`

Press **Enter** at the end of each entry. Each file name appears on a separate line.

Since these strings represent logical document locations and not actual file paths, only forward slashes are needed no matter the platform.

Changes to the exclusion list are picked up when the node agent is restarted.

Data type	String
Units	File names or patterns

z/OS location service daemons

Location service daemons provide the CORBA location service in support of Remote Method Invocation and Internet Inter-ORB Protocol (RMI/IIOP).

When a client makes a remote call to an enterprise bean, a location service daemon determines which servers are eligible to process the request. The location service daemon makes the decision with the z/OS workload management function (WLM). The daemon then routes the request to the selected server, which establishes a CORBA session with the client. Subsequent calls to the same enterprise bean flow directly over the established session.

In a cell, one location service daemon exists for each sysplex node group. A location service daemon process runs on each system that has a node in a sysplex node group. An example of a system is the z/OS operating system on a logical partition (LPAR).

Stopping or canceling the z/OS location service daemon from the MVS console

Location service daemons provide the CORBA location service in support of Remote Method Invocation and Internet Inter-ORB Protocol (RMI/IIOP). This topic discusses how to issue MVS console commands to stop or cancel the z/OS location service daemon.

You must first install WebSphere Application Server.

If you cancel or stop the location service daemon, it cancels or stops all WebSphere Application Servers on the system. If you installed Network Deployment, the location service daemon also cancels or stops the deployment manager and the node agents.

Issue one of the following commands to stop the location service daemon:

```
STOP DAEMON01
CANCEL DAEMON01
CANCEL DAEMON01,ARMRESTART
```

If you issue the **stop** command, the server finishes all remaining work before shutting itself down. If you issue the **cancel** command, the server stops quickly. Inflight data and transactions might be lost.

Use the **cancel** command that has the ARMRESTART option if automatic restart management (ARM) is active and you want the ARM to restart the location service daemon.

You have stopped or cancelled the location service daemon.

Determining if the z/OS location service daemon is running

This article describes what steps you must follow to determine if the location service daemon is running on a z/OS system.

The location service daemon starts automatically if it is not already running when you start an Application Server or a deployment manager server.

Perform the following step any time you want to know whether the location service daemon is running.

To determine if the location service daemon is running, issue one of the display commands as described in the Example: Displaying active address spaces topic in the *Using the administrative clients* PDF , such as `d a,l`, from the MVS console.

You know the system is running if you see the BBODMN*x* address space running, where *x* is a letter (A, B, C, and so on).

Modifying z/OS location service daemon settings

This article describes what steps you must follow to modify the location service daemon settings in the administrative console.

The location service daemon is an integral component of the Remote Method Invocation and Internet Inter-ORB Protocol (RMI/IIOP) communication function for the WebSphere Application Server for z/OS. The daemon works with z/OS workload management to distribute RMI requests (for example enterprise bean requests) among application servers in a cell.

Before you begin

Complete the following items before starting this task:

- Configure the location service daemon

  You must first configure the location service daemon in the WebSphere Application Server customization dialogs. After you configure the daemon, you can modify or view the location service daemon settings from the administrative console.
- Optionally enable security

  There is no specific setting to enable Secure Sockets Layer (SSL) for the location service daemon. If you want to use the SSL protocol to encrypt communications to the location service daemon, complete the following items:
  - Enable global security for the cell.
  - Define a valid system SSL repertoire for the location service daemon.
  - Set the z/OS user ID that is assigned to the location service daemon-started task to the same z/OS user ID that was used to create the key ring.

You can now modify the location service daemon settings in the administrative console.

1. Go to the Settings page for the location service daemon to view the help page.
2. Click **System Administration > Node groups > *sysplex node group* > z/OS Location Service Daemon** in the console navigation tree.
3. Specify the following values for the location service daemon:
   - The job name
   - The ports on which it listens
   - The IP names on which it listens
   - The start command
   - The start command arguments
   - The SSL repertoire that it uses

The location service daemon settings are modified.

z/OS location service daemon settings

In a cell, one location service daemon definition exists for each sysplex node group. A location service daemon process runs on each system that has a node in a sysplex node group in that cell. When a client makes a remote call to an enterprise bean, a location service daemon determines which server or servers are eligible to process the request, and routes the request to the selected server. An example of a system is the z/OS operating system on a logical partition (LPAR).

Changes made to these settings apply to the location service daemon in a sysplex node group.

This administrative console page is only visible if the node group contains z/OS nodes. To view this administrative console page, click **System Administration > Node groups > *sysplex node group* > z/OS Location Service Daemon**.

Job name

Specifies the job name of the location service daemon. This name can be from one to eight characters. You can use alphanumeric or the national language characters of @#$.

Data type	String
Default	None

Start command

Specifies the command string that servers use to automatically start the location service daemon.

Data type	String
Format	START *location service daemon JCL procedure name* The procedure name is defined at the node level by the customization dialog during customization. You can change the procedure name on the WebSphere Variables administrative console panels by going to **Environment > WebSphere Variables**.
Example	START BBO6DMN
Default	${NODE_DAEMON_START_COMMAND} NODE_DAEMON_START_COMMAND is the configuration variable name whose value you can change on the WebSphere Variables administrative console panels.

Listen IP name

Specifies the IP address on which the location service daemon listens. The Listen IP name must be either a dotted decimal IP address or an asterisk (*). The asterisk means that the location service daemon listens on all available IP addresses.

Data type	String
Default	asterisk (*)

Daemon IP name

Specifies the IP name that clients use to access enterprise beans and Common Object Request Broker Architecture (CORBA) components on servers that belong to the sysplex node group that this location service daemon serves.

The daemon IP name can be any of the following choices:

- IP name such as www.foobar.com
- IP address of the form n.n.n.n, where n is an integer
- Dynamic virtual IP address (DVIPA)

Data type	String
Default	None

Port

Specifies the port on which the location service daemon listens for Remote Method Invocation and Internet Inter-ORB (RMI/IIOP) requests.

Because the system reserves port numbers less than 1024 for Transmission Control Protocol/Internet Protocol (TCP/IP) applications, the recommendation is to use port numbers greater than 1023. However, the port range starts at 1.

Data type	Integer
Default	None
Range	1 to 65535

SSL port

Specifies the port on which the location service daemon listens for encrypted Remote Method Invocation and Internet Inter-ORB (RMI/IIOP) requests.

Because the system reserves port numbers less than 1024 for Transmission Control Protocol/Internet Protocol (TCP/IP) applications, the recommendation is to use port numbers greater than 1023. However, the port range starts at 0. A value of 0 disables the SSL port.

Data type	Integer
Default	None
Range	0 to 65535

SSL settings

Specifies a predefined list of Secure Sockets Layer settings for connections.

These settings are System SSL repertoires that you configured on the SSL repertoire panel. Select one repertoire from the list.

Administrative agents: Resources for learning

Use the following links to find relevant supplemental information about WebSphere Application Server administrative agents and distributed administration. 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:

Administration

- IBM WebSphere Application Server Redbooks

  This site contains a listing of all WebSphere Application Server Redbooks.
- IBM WebSphere developerWorks

  This site is the home of technical information for developers working with WebSphere products. You can download WebSphere software, take a fast path to developerWorks zones, such as VisualAge Java or WebSphere Application Server, learn about WebSphere products through a newcomers page, tutorials, technology previews, training, and Redbooks, get answers to questions about WebSphere products, and join the WebSphere community, where you can keep up with the latest developments and technical papers.
- WebSphere Application Server Support page

  Take advantage of the Web-based Support and Service resources from IBM to quickly find answers to your technical questions. You can easily access this extensive Web-based support through the IBM Software Support portal at URL `http://www-3.ibm.com/software/support/` and search by product category, or by product name. For example, if you are experiencing problems specific to WebSphere Application Server, click **WebSphere Application Server** in the product list. The WebSphere Application Server Support page appears.

Chapter 5. Configuring the environment

Use the following links to find relevant supplemental information about configuring the environment. The information resides on IBM and non-IBM internet sites, whose sponsors control the technical accuracy of the information.

To assist in handling requests among Web applications, Web containers, and application servers, you can configure cell-wide settings for virtual hosts, variables and shared libraries.

1. Configure virtual hosts.
2. Configure variables.
3. If your deployed applications use shared library files, define the shared library files needed.
   See "Managing shared libraries" on page 226.

Virtual hosts

When you configure WebSphere Application Server, you can associate a virtual host to one or more Web modules. Each Web module can be associated with one and only one virtual host.

A virtual host is a configuration entity that enables a single host machine to resemble multiple host machines. It maintains a list of Multipurpose Internet Mail Extensions (MIME) types that it processes. Resources associated with one virtual host cannot share data with resources associated with another virtual host, even if the virtual hosts share the same physical machine.

Each virtual host has a logical name and a list of one or more DNS aliases by which it is known. A DNS alias is the TCP/IP hostname and port number that is used to request the servlet, for example `yourHostName:80`. When no port number is specified, 80 is assumed.

A client request for a servlet, JavaServer Pages file, or related resource contains a DNS alias and a Uniform Resource Indicator (URI) that is unique to that resource. When a client request for a servlet, JavaServer Pages file, or related resource is received, the DNS alias is compared to the list of all known virtual host groups to locate the correct virtual host, and the URI is compared to the list of all known URI groups to locate the correct URI group. If the virtual host group and URI group are found, the request is sent to the corresponding server group for processing and a response is returned to browser. If a matching virtual host group or URI group is not found, an error is returned to the browser.

The first time that you start an application server, a default virtual host (named default_host) is configured. The DNS aliases for the default virtual host are configured as `*:80` and `*:9080`, where port 80 is the HTTP server port and port 9080 is the port for the default server's HTTP transport. The default virtual host includes common aliases, such as the machine's IP address, short host name, and fully qualified host name. One of these aliases comprises the first part of the path for accessing a resource such as a servlet. For example, the alias `localhost:80` is used in the request `http://localhost:80/myServlet.`

A virtual host is not associated with a particular node (machine). It is a configuration, rather than a live object, which is why you can create it, but cannot start or stop it. For many users, creating virtual hosts is unnecessary because the default_host is provided.

Adding a localhost to the virtual hosts adds the host name and IP address of the localhost machine to the alias table. This allows a remote user to access the administrative console.

You can use the administrative console to add or change DNS aliases if you want to use ports other than the default ports. If you do make a change to a DNS alias, you must regenerate the Web server plug-in configuration. You can use the administrative console to initiate the plug-in regeneration.

Note: You might want to add additional aliases or change the default aliases if:

- The HTTP server instance is running on a port other than 80. Add the correct port number to each of the aliases. For example, change `yourhost` to `yourhost:8000`.
- You want to make HTTPS requests, which use Secure Sockets Layer (SSL). To make HTTPS requests you must add port 443 to each of the aliases. Port 443 is the default port for SSL requests.
- Your Web server instance is listening for SSL requests on a port other than 443. In this situation, you must add that port number to each of the aliases.
- You want to use a port other then default port (9080) for the application server.
- You want to use other aliases that are not listed.

Why you would use virtual hosting

Virtual hosts let you manage a single application server on a single machine as if the application server were multiple application servers each on their own host machine. Resources associated with one virtual host cannot share data with resources associated with another virtual host. This is true even though the virtual hosts share the same application server on the same physical machine.

Virtual hosts isolate and independently manage multiple sets of resources on the same physical machine.

Suppose an Internet service provider (ISP) has two customers with Internet sites hosted on the same machine. The ISP keeps the two sites isolated from one another, despite their sharing a machine, by using virtual hosts. The ISP associates the resources of the first company with VirtualHost1 and the resources of the second company with VirtualHost2. Both virtual hosts map to the same application server.

Further suppose that both company sites offer the same servlet. Each site has its own instance of the servlet, and is unaware of the same servlet on the other site. If the company whose site is organized on VirtualHost2 is past due in paying its account with the ISP, the ISP can refuse all servlet requests that are routed to VirtualHost2. Even though the same servlet is available on VirtualHost1, the requests directed at VirtualHost2 do not go to the other virtual host.

The servlets on one virtual host do not share their context with the servlets on the other virtual host. Requests for the servlet on VirtualHost1 can continue as usual. This is true even though VirtualHost2 is refusing to fill requests for the servlet with the same name.

You associate a servlet or other application with a virtual host instead of the actual DNS address.

The default virtual host (default_host)

The product provides a default virtual host (named default_host).

The virtual host configuration uses wildcard entries with the ports for its virtual host entries.

- The default alias is *:80, using an internal port that is not secure.
- Aliases of the form *:9080 use the secure internal port.
- Aliases of the form *:9443 use the external port that is not secure.
- Aliases of the form *:443 use the secure external port.

Unless you specifically want to isolate resources from one another on the same node (physical machine), you probably do not need any virtual hosts in addition to the default host.

How requests map to virtual host aliases

Virtual hosts let you manage a single application server on a single machine as if the application server were multiple application servers that are each on their own host machine. Resources associated with one virtual host cannot share data with resources associated with another virtual host, even though the virtual hosts share the same application server on the same physical machine.

When you request a resource, WebSphere Application Server tries to map the request to an alias of a defined virtual host.

Mappings are both case sensitive and insensitive. For example, the portion `http://host:port/` is not case sensitive, but the URL that follows is case sensitive. The match must be alphanumerically exact. Also, different port numbers are treated as different aliases.

For example, the request `http://www.myhost.com/myservlet` maps successfully to `http://WWW.MYHOST.COM/myservlet` but not to `http://WWW.MYHOST.COM/MYSERVLET` or `Www.Myhost.Com/Myservlet`. In the latter two cases, these mappings fail because of case sensitivity. The request `http://www.myhost.com/myservlet` does not map successfully to `http://myhost/myservlet` or to `http://myhost:9876/myservlet`. These mappings fail because they are not alphanumerically correct.

You can use wildcard entries for aliases by port and specify that all valid host name and address combinations on a particular port map to a particular virtual host.

If you request a resource using an alias that cannot be mapped to an alias of a defined virtual host, you receive a 404 error in the browser that was used to issue the request. A message states that the virtual host could not be found.

Two sets of associations occur for virtual hosts. Application deployment associates an application with a virtual host. Virtual host definitions associate the network address of the machine and the HTTP transport or Web server port assignment of the application server with the virtual host. Looking at the flow from the Web client request for the snoop servlet, for example, the following actions occur:

1. The Web client asks for the snoop servlet: at Web address `http://www.some_host.some_company.com:9080/snoop`
2. The some_host machine has the 9080 port assigned to the standalone application server, server1.
3. server1 looks at the virtual host assignments to determine the virtual host that is assigned to the alias `some_host.some_company.com:9080`.
4. The application server finds that no explicit alias for that DNS string exists. However, a wild card assignment for host name * at port 9080 does exist. This is a match. The virtual host that defines the match is default_host.
5. The application server looks at the applications deployed on the default_host and finds the snoop servlet.
6. The application server serves the application to the Web client and the requester is able to use the snoop servlet.

You can have any number of aliases for a virtual host. You can even have overlapping aliases, such as:

Virtual host	Alias	Port
default_host	*	9080
	localhost	9080
	my_machine	9080
	my_machine.my_company.com	9080
	localhost	80

The Application Server looks for a match using the explicit address specified on the Web client address. However, it might resolve the match to any other alias that matches the pattern before matching the explicit address. Simply defining an alias first in the list of aliases does not guarantee the search order when WebSphere Application Server is looking for a matching alias.

A problem can occur if you use the same alias for two different virtual hosts. For example, assume that you installed the default application and the snoop servlet on the default_host. You also have another virtual host called the admin_host. However, you have not installed the default application or the snoop servlet on the admin_host.

Assume that you define overlapping aliases for both virtual hosts because you accidentally defined port 9080 for the admin_host instead of port 9060:

Virtual host	Alias	Port
default_host	*	9080
	localhost	9080
admin_host	*	9060
	my_machine.com	9080

Assume that a Web client request comes in for `http://my_machine.com:9080/snoop`.

If the application server matches the request against `*:9080`, the application is served from the default_host. If the application server matches the request to `my.machine.com:9080`, the application cannot be found. A 404 error occurs in the browser that issues the request. A message states that the virtual host could not be found.

This problem is the result of not finding the requested application in the first virtual host that has a matching alias. The correct way to code aliases is for the alias name on an incoming request to match only one virtual host in all of your virtual host definitions. If the URL can match more than one virtual host, you can see the problem just described.

Configuring virtual hosts

For configuration purposes, a virtual host enables WebSphere Application Server to treat multiple host machines or port numbers as a single logical host (virtual host). You can combine multiple host machines into a single virtual host or assign host machines to different virtual hosts, to separate and control which WebSphere Application Server resources are available for client requests.

If your external HTTP server configuration uses the default port, 9080, you do not have to perform these steps.

You must update the HTTP port numbers associated with the default virtual host. or define a new virtual host and associate it with the ports your HTTP server configuration uses if:

- Your external HTTP server configuration uses a port other than the default port of 9080, you must define the port that you are using.
- You are using the default HTTP port 9080, but the port is no longer defined. You must define port 9080.
- You have created multiple application servers as either stand-alone servers or cluster members, and these servers use the same virtual host. Because each server must be listening on a different port, you must define a virtual host alias for the HTTP port of each server.

If you define new virtual host aliases, identify the port values that the aliases use on the “Host alias settings” on page 207 page in the administrative console.

To create a new virtual host or change the configuration of an existing virtual host:

1. In the administrative console, click **Environment > Virtual Hosts**.
2. **Optional:** Create a new virtual host. If you create a new virtual host, a default set of 90 MIME entries are automatically created for that virtual host.
   a. In the administrative console, click **New**.

   b. Enter the name of the new virtual host and click **OK**. The new virtual host appears in the list of virtual hosts you can configure.
3. Select the virtual host whose configuration you want to change.
4. Under Additional Properties, click **Host Aliases**.
5. Create new host aliases or update existing host aliases to associate each of your HTTP port numbers with this virtual host.

   There must be a virtual host alias corresponding to each port your HTTP server configuration uses. There is one HTTP port associated with each Web container, and it is usually assigned to the virtual host named `default_host`. You can change the default assignment to any valid virtual host.

   The host aliases associated with the `default_host` virtual host are set to * when you install WebSphere Application Server. The * (an asterisk) indicates that the alias name does not have to be specified or that any name can be specified.

   When the URL for the application is entered into a Web browser, the port number is included. For example, if 9082 is the port number, the specified URL might look like the following:

   ```
   http://localhost:9082/wlm/SimpleServlet
   ```

   To create a new host alias:

   a. Click **New**.

   b. Specify a host alias name in the Host Name field and one of your HTTP ports in the Port field.

      You can specify * (an asterisk) for the alias name if you do not want to require the specification of the alias name or if you want to allow any name to be specified.

   c. Click **OK** and **Save** to save your configuration change.

   To update an existing host alias:

   a. Select an existing host alias name.

   b. Change the value specified in the Port field to one of your HTTP ports.

   c. Click **OK** and **Save** to save your configuration change.
6. **Optional:** Define a MIME object type and its file name extension if you require a MIME type other than the pre-defined types.

   a. For each needed MIME entry on the "MIME type collection" on page 208 page, click **New**.

   b. On the "MIME type settings" on page 208 page, specify a MIME type and extension.

   c. Click **OK** and **Save** to save your configuration change.
7. Regenerate the Web server plug-in configuration.

   a. **Click Servers > Web servers**, then select the appropriate Web server.

   b. Click **Generate Plug-in**, then click **Propagate Plug-in**.
8. Restart the application server.

Virtual host collection

Use this page to create and manage configurations that each let a single host machine resemble multiple host machines. Such configurations are known as *virtual hosts.*

To view this administrative console page, click **Environment > Virtual Hosts**.

Each virtual host has a logical name (which you define on this panel) and is known by its list of one or more domain name system (DNS) aliases. A DNS alias is the TCP/IP host name and port number used to request the servlet, for example yourHostName:80. (Port 80 is the default.)

You define one or more alias associations by clicking an existing virtual host or by adding a new virtual host.

When a servlet request is made, the server name and port number entered into the browser are compared to a list of all known aliases in an effort to locate the correct virtual host to serve the servlet. No match returns an error to the browser.

An application server profile provides a default virtual host with some common aliases, such as the internet protocol (IP) address, the DNS short host name, and the DNS fully qualified host name. The alias comprises the first part of the path for accessing a resource such as a servlet.

For example, the alias is localhost:80 in the request http://localhost:80/myServlet.

A virtual host is not associated with a particular profile or node (machine), but is associated with a particular server instead. It is a configuration, rather than a ″live object.″ You can create a virtual host, but you cannot start or stop it.

For many users, creating virtual hosts is unnecessary because the default_host that is provided is sufficient.

Adding the host name and IP address of the localhost machine to the alias table lets a remote user access the administrative console.

Resources associated with one virtual host cannot share data with resources associated with another virtual host, even if the virtual hosts share the same physical machine.

Name

Specifies a logical name for configuring Web applications to a particular host name. The default virtual host is suitable for most simple configurations.

Virtual hosts enable you to isolate, and independently manage, multiple sets of resources on the same physical machine. Determine whether you need a virtual host alias for each port associated with an HTTP transport channel or an HTTP transport. There must be a virtual host alias corresponding to each port used by an HTTP transport channel or an HTTP transport. There is one HTTP transport channel or HTTP transport associated with each Web container, and there is one Web container in each application server.

When you create a virtual host, a default set of 90 MIME entries is created for the virtual host.

You must create a virtual host for each HTTP port in the following cases:

- You use the internal HTTP transport with a port other than the default value of 9080, or for some reason the virtual host does not contain the usual entry for port 9080.
- You create multiple application servers (stand-alone servers, managed servers, or cluster members) that are using the same virtual host. Because each server must be listening on a different HTTP port, you need a virtual host alias for the HTTP port of each server.

Virtual host settings

Use this page to configure a virtual host instance.

To view this administrative console page, click **Environment > Virtual Hosts >** *virtual_host_name*.

Name:

Specifies a logical name for configuring Web applications to a particular host name. The default virtual host is suitable for most simple configurations.

Data type	String
Default	default_host

Host alias collection

Use this page to manage host name aliases defined for a virtual host. An alias is the DNS host name and port number that a client uses to form the URL request for a Web application resource.

To view this administrative console page, click **Environment** > **Virtual Hosts** > *virtual_host_name* > **Host Aliases**.

Host Name:

Specifies the IP address, DNS host name with domain name suffix, or just the DNS host name, used by a client to request a Web application resource (such as a servlet, JavaServer Pages (JSP) file, or HTML page). For example, the host alias name is `myhost` in a DNS name of `myhost:8080`.

The product provides a default virtual host (named default_host). The virtual host configuration uses the wildcard character * (asterisk) along with the port number for its virtual host entries. Unless you specifically want to isolate resources from one another on the same node (physical machine), you probably do not need any virtual hosts in addition to the default host.

Port:

Specifies the port for which the Web server has been configured to accept client requests. For example, the port assignment is *8080* in a DNS name of `myhost:8080`. A URL refers to this DNS as: http://*myhost:8080*/servlet/snoop.

Host alias settings:

Use this page to view and configure a host alias.

To view this administrative console page, click **Environment** > **Virtual Hosts** > *virtual_host_name* > **Host Aliases** > *host_alias_name*.

Host name:

Specifies the IP address, domain name system (DNS) host name with domain name suffix, or the DNS host name that clients use to request a Web application resource, such as a servlet, JSP file, or HTML page.

For example, when the DNS name is `myhost`, the host alias is `myhost:8080`, where *8080* is the port. A URL request can refer to the snoop servlet on the host alias as: http://*myhost:8080*/servlet/snoop.

When there is no port number specified for a host alias, the default port is 80. For existing virtual hosts, the default host name and port reflect the values specified at product installation or configuration. For new virtual hosts, the default can be `*` to allow any value or no specification.

Data type	String
Default	* You can also use the IP address or the long or short DNS name.

Port:

Specifies the port where the Web server accepts client requests. Specify a port value in conjunction with the host name.

The default reflects the value specified at product setup. The default might be `80`, `81`, `9060` or a similar value.

Data type	Integer
Default	9060

MIME type collection

Use this page to view and configure multi-purpose internet mail extensions (MIME) object types and their file name extensions.

The list shows a collection of MIME type extension mappings defined for the virtual host. Virtual host MIME entries apply when you do not specify MIME entries at the Web module level.

To view a list of current virtual host Mime types in the administrative console, click **Environment > Virtual Hosts >** *virtual_host_name* **> MIME Types**.

MIME Type:

Specifies a MIME type, which can be application, audio, image, text, video, www, or x-world. An example value for MIME type is `text/html`.

Extensions:

Specifies file extensions of files that map the MIME type. Do not specify the period before the extension. Example extensions for a `text/html` MIME type are `htm` and `html`.

MIME type settings:

Use this page to configure a multi-purpose internet mail extensions (MIME) object type.

To view this administrative console page, click **Environment > Virtual Hosts >** *virtual_host_name* **> MIME Types >** *MIME_type*.

MIME Type:

Specifies a MIME type, which can be application, audio, image, text, video, www, or x-world. An example value for MIME type is `text/html`.

An example value for MIME type is `text/html`. A default value appears only if you are viewing the configuration for an existing instance.

Data type	String

Extensions:

Specifies file extensions of files that map the MIME type. Do not specify the period before the extension. Example extensions for a `text/html` MIME type are `htm` and `html`.

File extensions for a `text/html` MIME type are `.htm` and `.html`. A default value appears only if you are viewing the configuration for an existing MIME type.

Data type	String

Variables

A variable is a configuration property that can be used to provide a parameter for some values in the system. A variable has a name and a value.

Not all WebSphere components support the use of a variable that you can define using this function. Test your application to verify that variables that you define are being used correctly.

WebSphere variables are used for:

- Configuring WebSphere Application Server path names, such as JAVA_HOME, and APP_INSTALL_ROOT.
- Configuring certain cell-wide or cluster-widecustomization values.
- Configuring the WebSphere Application Server for z/OS location service.

Each variable has a scope. A scope is the range of locations in the WebSphere Application Server network where the variable is applicable.

- A variable with a cell-wide scope is available across the entire deployment manager cell.
- A variable with a cluster-wide scope is available across the entire cluster in the cell.
- A variable with a node-level scope is available only on the node and the servers on that node. If a node-level variable has the same name as a cell-wide variable, the node-level variable value takes precedence.
- A server variable is available only on the one server process. A server variable takes precedence over a variable with the same name that is defined at a higher level.

You can use variables in configuration values such as file system path settings. Use the following syntax to refer to a variable:

```
${variable_name}
```

The value of a variable can contain a reference to another variable. The value of the variable is computed by substituting the value of the referenced variable recursively.

Variables are useful when concatenating two path variables when the specification does not accept the `AND` operator. For example, suppose that the following variables exist:

Variable name	**Variable value**
ROOT_DIR	`/`
HOME_DIR	`${ROOT_DIR}home`
USER_DIR	`${HOME_DIR}/myuserdir`

The variable reference `${USER_DIR}` resolves to the value `/home/myuserdir`.

Configuring WebSphere variables

This topic describes how to create a WebSphere Application Server variable.

You can define a WebSphere Application Server variable to provide a parameter for some values in the system. After you define the name and value for a variable, the value is used in place of the variable name. Variables most often specify file paths. However, some system components also support the use of variables. The Variable settings topic supplies further details about specifying variables and highlights further details about WebSphere Application Server components that use them.

WebSphere variables are used for:

- Configuring WebSphere Application Server path names, such as JAVA_HOME, and APP_INSTALL_ROOT.
- Configuring certain cell-wide or cluster-wide customization values.
- Configuring the WebSphere Application Server for z/OS location service.

The scope of a variable can be cell-wide, cluster-wide, node-wide, or applicable to only one server process.

Define variables on the **Environment > WebSphere Variables** console page.

Define the scope to apply a variable cell-wide, cluster-wide, node-wide, or to only one server process. A variable resolves to its new value when used in a component that supports the use of variables.

1. Click **Environment > WebSphere Variables** in the administrative console to define a new variable.
2. Specify the scope of the variable.

   Declare the new variable for the **Cell**, **Cluster**, **Node**, or **Server** and click **Apply**.

   The variable exists at the level you specify. Define a variable at multiple levels to use multiple values. The more granular definition overrides the higher level setting.

   For instance, if you specify the same variable on a node and a server, the server setting overrides the node setting. Similarly, a node level setting overrides a cluster or cell setting.

   Scoping variables is particularly important when testing data source objects. Variable scoping can cause a data source to fail the test connection, but to succeed at run time, or to pass the test connection, but fail at run time.

   See the *Administering applications and their environment* PDF for more information.
3. Click **New** on the WebSphere Variables page.
4. Specify a name, a value, and a description on the Variable page. Click **OK**. You can use the following options for setting WebSphere Application Server variables:
   - Use "WebSphere variables predefined for the z/OS platform" on page 214.
   - Use WebSphere Application Server variables to modify the daemon configuration. By appending a server custom property onto a daemon tag, you can designate that variable specifically for that daemon. Enter `DAEMON_<server custom property>` in the **Name** field. For example, if you enter `DAEMON_ras_trace_outputlocation` in the Name field and `SYSOUT` in the Value field, you can direct that particular daemon's trace output to SYSPRINT.
   - WebSphere Application Server variables support substitution. The name of a variable can be formed by substituting the value of another variable. If you enter `${<`*`variable name`*`>}` in the **Name** field, the value of `<`*`variable name`*`>` will be the name of your new WebSphere variable. For example `${JAVA_HOME}` will create a WebSphere variable with a name that is equal to the Java home directory.
   - The application server uses WebSphere Application Server internal variables for its own purposes. The prefixes that indicate that a variable is internal are `WAS_DAEMON_<`*`server custom property`*`>`, `WAS_DAEMON_ONLY_<`*`server custom property`*`>`, and `WAS_SERVER_ONLY_<`*`server custom property`*`>`. Any variables with these tags are not intended for your use. They are reserved exclusively for use by the server run time. Modifying these variables can cause unexpected errors.
5. Verify that the variable is displayed in the list of variables. The administrative console does not pick up typing errors. The variable is ignored if it is referred to incorrectly.
6. Save your configuration.
7. Stop the server and start the server again to put the variable configuration change into effect.

WebSphere variables collection

Use this page to view and change a list of substitution variables with their values and scope.

To view this administrative console page, click **Environment > WebSphere Variables**.

For information on a variable, click the variable and read the value in the **Description** field.

Name

Specifies the symbolic name for a WebSphere Application Server variable. For example, a variable name might represent a physical path or URL root used by WebSphere Application Server.

Value

Specifies the value that the symbolic name represents. For example, the value might be an absolute path value for a file or URL root.

Scope

Specifies the level at which a WebSphere Application Server variable is visible on the administrative console panel.

A resource can be visible in the administrative console collection table at the cell, cluster, node, or server scope.

Variable settings

Use this page to define the name and value of a WebSphere Application Server substitution variable.

To view this administrative console page, click **Environment > WebSphere Variables >** *WebSphere_variable_name*.

Name:

Specifies the symbolic name for a WebSphere Application Server variable. For example, a variable name might represent a physical path or URL root that is used by WebSphere Application Server.

WebSphere Application Server variables are used for:

- Configuring WebSphere Application Server path names, such as *JAVA_HOME*, and APP_INSTALL_ROOT.
- Configuring certain cell-wide customization values.
- Configuring the WebSphere Application Server for z/OS location service.

WebSphere Application Server substitutes the symbolic name wherever its value displays in the system.

For example, *JAVA_HOME* is the symbolic name representing the file system path to the installation directory for the Java virtual machine (JVM). For example, the value is /opt/IBM/WebSphere/AppServer/java for the WebSphere Application Server product on a Linux machine.

You can create new variables for use in WebSphere Application Server components that support the use of variables.

The WebSphere Application Server security component supports variables when establishing administrative security, but only the following:

- APP_INSTALL_ROOT
- WAS_INSTALL_ROOT
- USER_INSTALL_ROOT
- WAS_TEMP_DIR
- WAS_PROPS_DIR
- WAS_ETC_DIR

The security component uses these variables as security defaults when making substitutions that identify a path to the security configuration settings.

Data type	String

Value:

Specifies the value that the symbolic name represents. For example, the value might be an absolute path value for a file or URL root.

For example, `/opt/IBM/WebSphere/AppServer/java` is the value on a Linux machine for a variable named JAVA_HOME.

Data type	String

Description:

Documents the purpose of a variable.

Data type	String

Variables

Variables in the WebSphere Application Server environment come in many varieties. They are used to control settings and properties relating to the server environment. Three main variable options that are important for a WebSphere Application Server user to know and understand are environment variables, and WebSphere Application Server variables, and custom properties.

Environment variables. Environment variables, also called *native environment variables*, are not specific to WebSphere Application Server and are defined by other elements, such as UNIX, Language Environment (LE), or third-party vendors, among others. Some of the UNIX-specific native variables are LIBPATH and STEPLIB. These variables tend to be operating system-specific.

Environment variables can also be specified as a servant custom property. To specify an environment variable as a servant custom property, in the administrative console, click **Application Server >** *server_name* **> Java and Process Management > Process Definition > Servant > Environment Entries**. This path is also used to set environment variables that control the collection of application server and Web container information in z/OS System Management Facility (SMF) records.

WebSphere Application Server variables

WebSphere Application Server variables are used for three purposes:

- Configuring WebSphere Application Server path names, such as JAVA_HOME, and APP_INSTALL_ROOT
- Configuring certain cell-wide customization values
- Configuring the WebSphere Application Server for z/OS location service

WebSphere Application Server variables are specified in the administrative console by clicking **Environment > Manage WebSphere variables**. How the variable is set determines its scope.

A variable can apply to a cell, a cluster, a node, or a server.

If the variable is set:

- At the server level, it applies to the entire server.

- At the node level, it applies to all servers in the node, unless you set the same variable at the server level. In that case, for that server, the setting that is specified at the server level overrides the setting that is specified at the node level.
- At the cell level, it applies to all nodes in that cell, unless you set the same variable at the node or server level.
  – If you set the same variable at the server level, for that server, the setting that is specified at the server level overrides the setting that is specified at the cell level.
  – If you set the same variable at the node level, for all servers in that node, the setting that is specified at the node level overrides the setting that is specified at the cell level.

Custom properties

Custom properties are property settings meant for a specific functional component. Any configuration element can have a custom property. Common configuration elements are cell, node, server, Web container, and transaction service. A limited number of supported custom properties are available and these properties can be set in the administrative console using the custom properties link that is associated with the functional component.

For example, to set Web container custom properties, click **Servers > Application Servers >** *server_name* **> Web Container Settings > Web container > Custom Properties**

Custom properties set from the Web container custom properties page apply to all transports that are associated with that Web container; custom properties set from one of the Web container transport chain or HTTP transport custom properties pages apply only to that specific HTTP transport chain or HTTP transport. If the same property is set on both the Web container page and either a transport chain or HTTP transport page, the settings on the transport chain or HTTP transport page override the settings that are defined for the Web container for that specific transport.

IBM Toolbox for Java JDBC driver

WebSphere Application Server supports the **IBM Toolbox for Java** JDBC driver. The IBM Toolbox for Java JDBC driver is included with the IBM Toolbox for Java product.

IBM Toolbox for Java is a library of Java classes that are optimized for accessing iSeries and AS/400 data and resources. You can use the IBM Toolbox for Java JDBC driver to access local or remote **DB2 UDB for iSeries** databases from server-side and client Java applications that run on any platform that supports Java.

IBM Toolbox for Java is available in these versions:

IBM Toolbox for Java licensed program

The licensed program is available with every OS/400 release, starting with Version 4 Release 2 (V4R2). You can install the licensed program on your iSeries system, and then either copy the IBM Toolbox for Java JAR file (*jt400.jar*) to your system or update your system *classpath* to locate the server installation. Product documentation for IBM Toolbox for Java is available from the iSeries information center: http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/index.htm Locate the documentation by traversing the following path in the left-hand navigation window of the iSeries information center: **Programming** > **Java** > **IBM Toolbox for Java**.

JTOpen

JTOpen is the open source version of IBM Toolbox for Java, and is more frequently updated than the licensed program version. You can download JTOpen from http://www-1.ibm.com/servers/eserver/iseries/toolbox/downloads.htm. You can also download the *JTOpen Programming Guide*. The guide includes instructions for installing JTOpen and information about the JDBC driver.

The JDBC driver for both versions supports JDBC 3.0. For more information about IBM Toolbox for Java and JTOpen, see the product Web site at http://www-1.ibm.com/servers/eserver/iseries/toolbox/index.html.

Note: If you are using WebSphere Application Server on platforms other than iSeries, use the **JTOpen** version of the Toolbox JDBC driver.

Configure and use the jt400.jar file

The following steps describe how to configure and utilize the jt400.jar file.

1. Download the *jt400.jar* file from the **JTOpen** URL at http://www-1.ibm.com/servers/eserver/iseries/toolbox/downloads.htm.

   Place it in a directory on your workstation such as */JDBC_Drivers/Toolbox*.
2. Open the administrative console.
3. Select **Environment > WebSphere Variables**.
4. Set the WebSphere variable *OS400_TOOLBOX_JDBC_DRIVER_PATH* at the **Node** level.
5. Double click **OS400_TOOLBOX_JDBC_DRIVER_PATH**.
6. Set the value to the full directory path to the `jt400.jar` file downloaded in step one. Do not include *jt400.jar* in this value.

   For example:

   ```
   OS400_TOOLBOX_JDBC_DRIVER_PATH == "/JDBC_Drivers/Toolbox"
   ```

   When you choose a Toolbox driver from the list of possible resource providers the **Classpath** field looks like:

   ```
   Classpath == ${OS400_TOOLBOX_JDBC_DRIVER_PATH}/jt400.jar
   ```

WebSphere variables predefined for the z/OS platform

The following WebSphere variables are predefined for the z/OS platform.

protocol_https_cert_mapping_file

This variable can be set at the cell, node, or server level. It specifies the name of a file containing entries that map IP addresses to server certificate labels. When an HTTP SSL connection request is received, the application server checks the IP address against entries in the file specified on this variable. If a match is made, the certificate mapped to the IP address is used for the connection. If no match is found, the application server checks for the existence of the `protocol_https_default_cert_label` variable. If a value has been specified on the `protocol_https_default_cert_label` variable, the certificate specified will be used to establish the connection. If no value has been specified on the `protocol_https_default_cert_label` variable, the default server certificate specified in the RACF SSL keyring owned by the application server will be used to establish the HTTP SSL connection.

Important: The protocol_https_cert_mapping_file variable is deprecated. You should use a workload classification document instead of a transaction class mapping file to classify work requests in a z/OS environment.

To set this variable at the cell level, in the administrative console click **Environment > WebSphere Variables > New** and enter `protocol_https_cert_mapping_file` in the Name field and the appropriate file name in the Value field.

To set this variable at the node level, in the administrative console click **System administration > Nodes** > *node_name* > **Custom properties > New** and enter `protocol_https_cert_mapping_file` in the Name field and the appropriate file name in the Value field.

To set this variable at the server level, in the administrative console click **Servers > Application servers >** *server_name* > **Web container settings > Web container > Custom properties > New** and enter `protocol_https_cert_mapping_file` in the Name field and the appropriate file name in the Value field.

Data type	String

protocol_https_default_cert_label

This variable can be set at the cell, node, or server level. It specifies the label of the server certificate that should be used when establishing HTTP SSL connections with the application server. If no value is specified and the IP address of the server does not match an IP address contained in the certificate mapping file specified on the `protocol_https_cert_mapping_file` variable, the default server certificate specified in the RACF SSL Keyring owned by the application server will be used to establish the HTTP SSL connection.

Important: The protocol_https_default_cert_label variable is deprecated. You should use a workload classification document instead of a transaction class mapping file to classify work requests in a z/OS environment.

To set this variable at the cell level, in the administrative console click **Environment** > **WebSphere Variables** > **New** and enter `protocol_https_default_cert_label` in the Name field and the appropriate certificate label in the Value field.

To set this variable at the node level, in the administrative console click **System administration** > **Nodes** > *node_name* > **Custom properties** > **New** and enter `protocol_https_default_cert_label` in the Name field and the appropriate certificate label in the Value field.

To set this variable at the server level, in the administrative console click **Servers** > **Application servers** > *server_name* > **Web container settings** > **Web container** > **Custom properties** > **New** and enter `protocol_https_default_cert_label` in the Name field and the appropriate certificate label in the Value field.

Data type	String

wlm_classification_file

This variable can be set at the cell, node, or server level. It specifies the location of your workload classification document.

Use this variable when classifying z/OS workload using the common XML file for classifying inbound HTTP, IIOP, and message-driven bean (MDB) work as described in “Classifying z/OS workload” on page 381. Any rules that are defined in the common XML file override the old format HTTP classification that is described in the article, “Transaction class mapping file entries” on page 396. The common XML file also overrides any rules in the MDB classification file that is defined by the endpoint_config_file WebSphere variable.

For example, your configuration has the common XML file defined in a server that also has the old format HTTP classification document specified. There are no HTTP classification rules defined in your common XML file, so the old format file is used to classify inbound HTTP requests. However, if your common XML file contains HTTP rules, these classification rules are used instead of classification rules that you defined in the old style HTTP classification document.

To set this variable at the cell level, in the administrative console click **Environment** > **WebSphere Variables** > **New** and enter `wlm_classification_file` in the Name field and the location of your workload classification document in the Value field.

To set this variable at the node level, in the administrative console click **System administration** > **Nodes** > *node_name* > **Custom properties** > **New** and enter `wlm_classification_file` in the Name field and the location of your workload classification document in the Value field.

To set this variable at the server level, in the administrative console click **Servers** > **Application servers** > *server_name* > **Web container settings** > **Web container** > **Custom properties** > **New** and enter `wlm_classification_file` in the Name field and the location of your workload classification document in the Value field.

Data type	String

Certificate mapping file entries

The following is the syntax for entries in a certificate mapping file.

```
SSLServerCert label ipaddress
```

where:

label Is the label of the server certificate in single or double quotes. If the label itself contains a single quote, double quotes are required as the delimiter.

ipaddress
Is the IP address of the server from which the request was received.

Examples:

```
SSLServerCert 'My Certificate Label' 9.57.4.29

SSLServerCert "My Co.'s Certificate" 9.57.4.30
```

Repository service custom properties on z/OS

Use this page to add custom properties for the repository service.

You can specify repository service custom properties in the administrative console:

1. In the administrative console navigation, click **System Administration** > **Node Agents**
2. Select a node agent from the list.
3. Under Additional Properties, click **File Synchronization Service**.
4. Under Additional Properties, click **Custom Properties**.
5. Click **New**.
6. Enter the name of the custom property in the Name field, and the value in the Value field. You can leave the Description field blank.

Support for the following custom property is provided with the WebSphere Application Server for z/OS product.

recoveryNode

Specifies that a node is a recovery node. Set this value to `true` if you want a node in a Network Deployment cell to act as a peer restart and recovery node for another node in the same cell. The recovery node shadows the complete configuration of its recovery peer. Use this property if you need to support peer restart and recovery only and are not using a shared file system.

Data type	Boolean

Application server z/OS custom properties

Use this page to configure WebSphere Application Server custom properties on a z/OS platform.

Application server custom properties can be specified in the administrative console:

- For an application server, click **Application servers** > *server1* > **Server infrastructure** > **Administration** > **Custom Properties**.
- For a deployment manager, click **System administration > Deployment manager**, and then under Server Infrastructure, click > **Java and Process Management > Process Definition > Java Virtual Machine > Custom Properties**.

Support for the following server custom properties is provided with the WebSphere Application Server for z/OS product.

Note: Setting server properties as server-level WebSphere Application Server for z/OS variables is deprecated. However, you can set a WebSphere Application Server for z/OS variable with a scope of `cell` or `node` to establish custom defaults for server properties.

control_region_dreg_on_no_srs

Specifies whether or not the controller rejects requests for dispatch within a servant when it detects that there are no servants available to process requests.

If this property is set to `1`, when the controller detects that there are no servants available to process requests, it rejects requests for dispatch within the servants, de-registers the application server with WLM, and stops the HTTP and MDB listeners. If this property is set to `0` (zero) the function is disabled.

When a minimal number of servants become available again, the controller re-registers the application server with WLM, starts the HTTP and MDB listeners, and allows requests to be dispatched to the servants.

Data Type	Integer
Acceptable values	0 or 1
Default	0

control_region_confirm_recovery_on_no_srs

Indicates when requests should be dispatched to servants following the detection of a no-servants situation. This property is ignored if the control_region_dreg_on_no_srs custom property is set to `0`.

If this property is set to `1`, requests are not dispatched to the servants until after the WebSphere Application Server administrator responds to WTOR message BBOO0297A. This message is issued following a no-servant situation when the sever detects that the required minimal number of servants are available to process requests.

If this property is set to `0` (zero), the controller determines when to allow requests to be dispatched to the servants after a no-servant condition is detected.

Data Type	Integer
Acceptable values	0 or 1
Default	0

control_region_timeout_delay

Specifies the number of seconds a controller waits after detecting a timeout before it terminates the servant. This time delay gives work that is currently running in the servant a chance to complete before the servant is terminated. When a servant thread completes work and sees that the servant is being terminated, the servant thread waits instead of selecting a new piece of work.

When this field is set to `0` (zero) the controller terminates a servant as soon as the controller detects a timeout.

Data Type	Integer
Units	Seconds
Default	0

control_region_mdb_request_timeout

Specifies the time, in seconds, that the server waits for a message-driven bean (MDB) request to receive a response. If the response is not received within the specified amount of time, the server removes the MDB request.

Set this value to `0` (zero) to disable the function.

Data Type	Integer
Units	Seconds
Default	120

protocol_accept_http_work_after_min_srs

Specifies whether or not the application server waits for a minimum number of servants (specified on the wlm_minimumSRCount variable) to be up before starting the HTTP transports. If this property is set to `true`, when the minimum number of servants is ready for work, the HTTP transport starts accepting work. If this property is set to `false`, the HTTP transports are started when the controller starts.

Data Type	Boolean
Default	false
Used by Daemon	no

protocol_bboc_log_response_failure

Specifies that if the BBOO0168W message is issued, the failure detected when attempting to send a response to a client is recorded. The message is sent to the error log. The message text contains the request method name, the reply status, and routing information identifying the client.

Data Type	Boolean
Default	false
Used by Daemon	yes

protocol_bboc_log_return_exception

Used to indicate that if the BBOO0169W message is issued, the response that contains the SystemException is recorded. The message is sent to the error log. The message text contains the exception identifier and minor code, the request method name, and routing information identifying the client.

Data Type	Boolean
Default	false
Used by Daemon	yes

protocol_giop_level_highest

Specifies the CORBA General Inter-ORB Protocol (GIOP) protocol version level that is used by the application server object request broker (ORB). Valid values are 1.1 and 1.2. Interoperable object references (IORs) exported from this server use the GIOP level indicated.

You might need to change this property from the default if you use a non-WebSphere Application Server client ORB that supports a lower version of the CORBA standard. For example, you might need to change from the default protocol version level of 1.2 to 1.1 to support an older, non-WebSphere Application Server client ORB.

Data Type	String
Default	1.2
Used by Daemon	Yes

protocol_http_backlog

Specifies the maximum length for the queue of pending connections using HTTP. The value used can be limited by the specification of the SOMAXCONN statement in the TCP/IP profile.

Data Type	Integer
Default	10
Used by Daemon	No

protocol_http_large_data_response_buffer

Specifies, in bytes, the maximum length of the response buffer used for HTTP requests. Responses larger than this value are rejected. A value of 0 indicates not to allocate a large response buffer. An HTTP large response buffer is not required if all the HTTP responses are less than 10 MB.

Data Type	Integer
Default	104857600
Used by Daemon	No

protocol_http_large_data_inbound_buffer

Specifies, in bytes, the length of a serially reusable inbound buffer used for HTTP requests larger than 10 megabytes. This value limits the size of incoming requests. For example, if you set the property to 15 megabytes, any requests over 15 megabytes are rejected. Specify `0` (zero) to indicate that no buffer is needed. Requests that are larger than 10 megabytes are rejected.

Data Type	Integer
Default	0
Used by Daemon	No

protocol_http_timeout_output_recovery

Specifies the action for the timer expiration. Set the value to SESSION to send the client a message stating that the server timed out and let the server continue running.

Data Type	String
Default	SERVANT
Used by Daemon	No

protocol_https_backlog

Specifies the maximum length for the queue of pending connections using HTTPS. The value used can be limited by the specification of the SOMAXCONN statement in the TCP/IP Profile.

Data Type	Integer
Default	10
Used by Daemon	No

protocol_iiop_backlog

Used to specify the maximum length for the queue of pending connections using the CORBA Internet Inter-ORB protocol (IIOP). The value used may be limited by the specification of the SOMAXCONN statement in the TCP/IP profile.

Data Type	Integer
Default	10
Used by Daemon	Yes

protocol_iiop_backlog_ssl

Used to specify the maximum length for the queue of pending connections using IIOP Secure Sockets Layer (SSL). The value used can be limited by the specification of the SOMAXCONN statement in the TCP/IP Profile.

Data Type	Integer
Default	10
Used by Daemon	Yes

protocol_iiop_resolve_foreign_hostname

Specifies whether to perform DNS resolution of the IP address of a foreign client to a DNS registered hostname for each established IIOP session. If this property is set to 1, the DNS hostname resolution is performed. If this property is set to 0, the DNS hostname resolution is not performed, and a textual representation of the IP address of the foreign client is used instead of the DNS hostname.

Data Type	boolean
Default	1
Used by Daemon	Yes

ras_debugEnabled

Specifies to use an external debugger tool with the application server for tracing and debugging client and server application components such as JavaServer Pages files, servlets, and enterprise beans.

Data Type	Boolean
Default	false
Used by Daemon	Yes

ras_default_msg_dd

Specifies whether to redirect write-to-operator (WTO) messages that use the default routing to hardcopy. These messages are redirected to the location identified through the DD card on the server's job control language (JCL) start procedure. These write to operator (WTO) messages are primarily messages that WebSphere Application Server for z/OS issues during initialization.

Data Type	String
Default	empty string
Used by Daemon	Yes

ras_dumpoptions_dumptype

Specifies the default dump that is used by the signal handler. Do not change this property unless directed by IBM service personnel.

0	No dump is generated.
1	A ctrace dump is taken.
2	A cdump dump is taken.
3	A csnap dump is taken.
4	A CEE3DMP dump is taken.

Data Type	Integer
Default	3
Used by Daemon	Yes

ras_dumpoptions_ledumpoptions

Specifies dump options use with a CEE3DMP dump. If you want more than one option, separate each option with a blank. Do not change this property unless directed by IBM service personnel.

Data Type	String
Default	THREAD(ALL) BLOCKS
Used by Daemon	Yes

ras_hardcopy_msg_dd

Specifies redirect write to operator (WTO) messages that WebSphere Application Server for z/OS routes to hardcopy. These messages are redirected to the location that is identified through the DD card on the server JCL start procedure. These WTO messages are primarily audit messages issued from Java code during initialization.

Data Type	String
Default	empty string
Used by Daemon	Yes

ras_log_logstreamName

Specifies the log stream for WebSphere Application Server for z/OS to use for error information. If the specified log stream is not found or not accessible, a message is issued and errors are written to the server job log. If this variable is not specified, WebSphere Application Server for z/OS uses the STDERR stream.

Data Type	String
Default	empty string
Used by Daemon	Yes

ras_minorcode_action

Specifies the default behavior for gathering documentation about system exception minor codes.

Data Type	String
Default	NODIAGNOSTICDATA
Used by Daemon	Yes

You can also specify:

- `CEEDUMP` - Captures callback and offsets. It takes time for the system to take CEEDUMPs. Transaction time-outs can occur.
- `TRACEBACK` - Captures Language Environment and z/OS UNIX traceback data.
- `SVCDUMP` - Captures an MVS dump (but does not produce a dump in the client).

ras_time_local

Specifies whether timestamps in the error log display is in local time or Greenwich Mean Time (GMT). The timestamp is in GMT if this property is set to false.

Data Type	Boolean
Default	false
Used by Daemon	Yes

ras_trace_basic

Specifies tracing overrides for particular WebSphere Application Server for z/OS subcomponents. Subcomponents, specified by numbers, receive basic and exception traces. If you specify more than one subcomponent, use parentheses and separate the numbers with commas. Contact IBM service for the subcomponent numbers and their meanings. Do not change this property unless directed by IBM service personnel.

Data Type	String
Default	null (empty string)
Used by Daemon	Yes

ras_trace_BufferCount

Specifies the number of trace buffers to allocate.

Data Type	Integer
Valid values	4 through 8
Default	4
Used by Daemon	Yes

ras_trace_BufferSize

Specifies the size of a single trace buffer in bytes. You can use the letters `K`(for kilobytes) or `M` (for megabytes).

Data Type	String
Valid values	128K through 4M
Default	1M
Used by Daemon	Yes

ras_trace_ctraceParms

Specifies the identify of the CTRACE PARMLIB member. The value can be either a two-character suffix, which is added to the CTIBBO string to form the name of the PARMLIB member, or the fully specified name of the PARMLIB member. For example, you can use the 01 suffix, which the system resolves to CTIBBO01. A fully specified name must conform to the naming requirements for a CTRACE PARMLIB member. For details, see *z/OS MVS Diagnosis: Tools and Service Aids*, GA22-7589.

If this property is specified and the PARMLIB member is not found, the default PARMLIB member, CTIBBO00, is used. If neither the specified nor the default PARMLIB member is found, tracing is defined to CTRACE, but no connection is available to a CTRACE external writer.

Data Type	String
Default	null (empty string)
Used by Daemon	Yes

ras_trace_defaultTracingLevel

Specifies the default tracing level for WebSphere Application Server for z/OS. Use this variable with the ras_trace_basic and ras_trace_detail variables to set tracing levels for Application Server for z/OS subcomponents. Do not change this property unless directed by IBM Support personnel.

0	No tracing
1	Exception tracing
2	Basic and exception tracing
3	Detailed tracing, including basic and exception tracing

Data Type	Integer
Default	1
Used by Daemon	Yes

ras_trace_detail

Specifies tracing overrides for particular WebSphere Application Server for z/OS subcomponents. Subcomponents, specified by numbers, receive detailed traces. If you specify more than one subcomponent, use parentheses and separate the numbers with commas. Contact IBM service for the subcomponent numbers and their meanings. Do not change this property unless directed by IBM Support personnel.

Data Type	String
Default	null (empty string)
Used by Daemon	Yes

ras_trace_exclude_specific

Specifies WebSphere Application Server for z/OS trace points to exclude from tracing activity.

Trace points are specified by 8-digit, hexadecimal numbers. Do not use this property unless directed by IBM service personnel. If IBM service personnel directs you to specify more than one trace point, use parentheses and separate the numbers with commas. You also can specify a WebSphere Application Server for z/OS variable name by enclosing the name in single quotes.

Data Type	String
Default	empty string
Used by Daemon	Yes

Note: Sometimes results depend on the ras_trace_minorCodeDefault environment variable. If you code `ras_trace_minorCodeTraceBacks=ALL` and `ras_minorcode_action=NODIAGNOSTICDATA`, you get a traceback. But, if you code `ras_trace_minorCodeTraceBacks=(null value)` and `ras_minorcode_action=TRACEBACK`, you also get a traceback. So, specifying `ras_trace_minorCodeTraceBacks=(null value)` does not cancel the traceback; it does not cause TRACEBACK data to be collected.

ras_trace_outputLocation

Specifies where to send trace records.

- To SYSPRINT
- To a memory buffer (BUFFER), the contents of which are later written to a CTRACE data set
- To a trace data set (TRCFILE) that is specified on the TRCFILE DD statement in the server start procedure.

For servers, you might specify one or more values, separated by a space.

Data Type	String
Default	SYSPRINT BUFFER
Used by Daemon	Yes

ras_trace_specific

Specifies tracing overrides for specific WebSphere Application Server for z/OS trace points. Trace points are indicated by 8-digit, hexadecimal numbers. To specify more than one trace point, use parentheses and separate the numbers with commas. You can also specify tracing on a specific environment variable by using the name enclosed in single quotes. Do not use this property unless directed by IBMSupport personnel.

Data Type	String
Default	null (empty string)
Used by Daemon	Yes

security_SMF_record_first_auth_user

Specifies whether to record the first authenticated user under request dispatch in the SM120CRE field in the SMF server activity record.

If this property is set to 1, the first authenticated user under request dispatch is written to the SM120CRE field. If this property is set to 0 (default), the ID of the user under which the server activity began is written to the SM120CRE field.

Data Type	Boolean
Default	0
Used by Daemon	No

server_region_jvm_localrefs

Do not use this property unless directed by IBM Support personnel.

Data Type	Integer
Default	128
Used by Daemon	No

server_region_jvm_logfile

Specifies the Hierarchical File System (HFS) file in which Java Native Interface (JNI) and class debug messages from the Java virtual machine (JVM) are logged. Use this variable only in a single-server environment. If you use this property in a multiple-server environment, all of the servers write to the same file, so you might have difficulty using the file for diagnostic purposes.

Data Type	String (file name)
Default	empty string (no file name)
Used by Daemon	No

server_region_recycle_count

Specifies the number of transactions processed by a servant process after which the servant process is recycled. z/OS Workload management (WLM) ends the servant after all affinity requirements have been met. Specify a nonzero value to enable recycling.

You might want to enable recycling if, after running for an extended period of time, your application is experiencing out-of-memory exceptions . (Out-of-memory exceptions can result from memory leakage by your application.)

Data Type	Integer
Default	0
Used by Daemon	No

server_start_wait_for_initialization_Timeout

Specifies how long the startServer.sh command processing waits for WebSphere Application Server initialization to complete. By default, it waits indefinitely until initialization is complete.

You might want to use this property if you want to:

- Control how long the application server waits for other dependent servers to start.
- Limit the amount of wait time when trying to debug problems with application initialization. (For example, you might not want to continue waiting if auto-started Web applications unexpectedly enter a long wait.)

Data Type	Integer
Default	0

Used by Daemon	No

server_use_wlm_to_queue_work

Specifies whether or not WLM for z/OS is used for workload queuing.

This property should be set to 1 if you are using a stateless application models. With these models, application objects, such as Enterprise JavaBeans (EJBs) and HTTPSessions, are only resident in memory for the life of an individual request. Therefore, the WebSphere Application Server should be configured to enable WLM for z/OS to dynamically balance individual requests. This configuration allows the WebSphere Application Server to provide linear scalability and consistent, repeatable response times.

This property should be set to `0` (zero) if you are using conversational application models. With these models, a client might hold, and periodically interact with, a reference to a stateful object which is pinned in the memory of one of the WebSphere Application Server JVMs for a period of time that is greater than the duration of an individual request. For example, the client might be using HTTP sessions, stateful session beans or entity beans that are maintained in memory instead of being stored in a database or file system between requests, as is done in stateless application models.

These application models prevent WebSphere Application Server from enabling the use of WLM for z/OS for dynamic workload routing of individual requests in a clustered environment because of the client's affinity to a specific Java Virtual Machine (JVM). In this situation, a round robin algorithm should be used to handle the client's initial request. This algorithm evenly distributes creation of long term affinities and is the best technique for achieving a balanced utilization of system resources under these conditions.

If you set this property to 0 for conversational application models, you must also set the server_work_distribution_algorithm property to `1`.

If you want to exploit WLM for z/OS's round robin capability instead of the previously described WebSphere Application Server capability, see"Workload management (WLM) for z/OS" on page 373. The differences between WLM for z/OS's round robin capability and the WebSphere Application Server round robin capability is explained in the following scenario.

Scenario: A customer starts two clients to talk to a server. The server has two servants and each servant has multiple threads. The customer expects one client to go to one servant, and the 2nd client to go to the other servant. The WebSphere Application Server provided round robin, initiated by setting server_use_wlm_to_queue_work=0 and server_work_distribution_algorithm=1, acts this way. However, the round robin WLM for z/OS provides goes through all of the threads in the first servant before moving to the threads in the next servant.

When the server_use_wlm_to_queue_work property is set to 0 (zero), the wlm_minimumSRCount and wlm_maximumSRCount properties should be set to the same value. Because the work is not going thru WLM for z/OS, WLM or z/OS only starts the number of servants specified for the wlm_minimumSRCount property.

0 (zero)	WLM for z/OS is not used.
1	WLM for z/OS is used.
Default	1
Used by Daemon	No

server_work_distribution_algorithm

Specifies the type of work distribution algorithm the Application Server will use for workload balancing. This property is only used if the server_use_wlm_to_queue_work property is set to `0`. If the server_use_wlm_to_queue_work property is set to `1`, the value specified for this property is ignored.

0 (zero)	The hot thread algorithm is used. This option is not recommended.
1	The round robin algorithm is used. This value must be specified if the server_use_wlm_to_queue_work custom property is set to `0` for conversational application models.
Default	0
Used by Daemon	No

transaction_recoveryTimeout

Specifies the time, in minutes, that this controller (region) uses to attempt to complete all restarted transactions before issuing a write-to-operator-with-reply (WTOR) message to the console, requesting whether to:

- Stop trying to resolve all restart transactions
- Write transaction-related information to the job log or hard copy log
- Terminate

If the operator replies to continue the recovery, the controller (region) attempts recovery for the specified amount of time before reissuing the write-to-operator message. After all the transactions are resolved, the controller terminates. This variable applies only to controllers in peer restart and recovery mode.

Data Type	Integer
Default	15
Used by Daemon	No

Shared library files

Shared library files in WebSphere Application Server consist of a symbolic name, a Java class path, and a native path for loading Java Native Interface (JNI) libraries.

You can define a shared library at the cell, node, or server level. Defining a library at one of the three levels does not cause the library to be placed into the application server's class loader. You must associate the library to an application or server in order for the classes represented by the shared library to be loaded in either a server-wide or application-specific class loader.

A separate class loader is used for shared libraries that are associated with an application server. This class loader is the parent of the application class loader, and the WebSphere Application Server extensions class loader is its parent. Shared libraries that are associated with an application are loaded by the application class loader.

Managing shared libraries

Shared libraries are files used by multiple applications. Using the administrative console, you can define a shared library at the cell, node, or server level. You can then associate the library to an application, module or server to load the classes represented by the shared library in either a server-wide or application-specific class loader. Using an installed optional package, you can associate a shared library to an application by declaring the dependent library `.jar` file in the `MANIFEST.MF` file of the application. Refer to the Java 2 Platform, Enterprise Edition (J2EE) 1.4 specification, section 8.2 for an example.

If your deployed applications use shared library files, define shared libraries for the library files and associate the libraries with specific applications or modules or with an application server. Associating a shared library file with a server associates the file with all applications on the server. Use the Shared Libraries page to define new shared library files to the system and remove them.

- Use the administrative console to define a shared library.

   1. Create a shared library for each library file that your applications need.
   2. Associate each shared library with an application or module.
      – Associate a shared library with an application or module that uses the shared library file.
      – Associate a shared library with an application server so every application on the server can use the shared library file.
- Use an installed optional package to declare a shared library for an application.
- Remove a shared library.
   1. Click **Environment > Shared Libraries** in the console navigation tree to access the Shared Libraries page.
   2. Select the library to be removed.
   3. Click **Delete**.

   The list of shared libraries is refreshed. The library file no longer displays in the list.

Creating shared libraries

Shared libraries are files used by multiple applications.

The first step for making a library file available to multiple applications deployed on a server is to create a shared library for each library file that your applications need. When you create the shared libraries, set variables for the library files.

Use the Shared Libraries page to create and configure shared libraries.

1. Go to the Shared Libraries page.

   Click **Environment > Shared Libraries** in the console navigation tree.
2. Change the scope of the collection table to see what shared libraries are in a cell, node, server, or cluster.
   a. Select a cell, node, server, or cluster.
   b. Click **Apply**.
3. Click **New**.
4. Configure the shared library.
   a. On the settings page for a shared library, specify the name, class path, and any other variables for the library file that are needed.

      If the shared library specifies a native library path, refer to “Configuring native libraries in shared libraries.”
   b. Click **Apply**.
5. Repeat steps 1 through 4 until you define a shared library instance for each library file that your applications need.

Using the administrative console, associate your shared libraries with specific applications or modules or with the class loader of an application server. Associating a shared library file with a server class loader associates the file with all applications on the server.

Alternatively, you can use an installed optional package to associate your shared libraries with an application.

Configuring native libraries in shared libraries

Native libraries are platform-specific library files, including `.dll`, `.so`, or *SRVPGM objects, that can be configured within shared libraries. Native libraries are visible to an application class loader whenever the shared library is associated with an application. Similarly, native libraries are visible to an application server class loader whenever the shared library is associated with an application server.

When designing a shared library, consider the following conditions regarding Java native library support:

- The Java virtual machine (JVM) allows only one class loader to load a particular native library.
- There is no application programming interface (API) to unload a native library from a class loader.

  Native libraries are unloaded by the JVM when the class loader that found the library is collected from the heap during garbage collection.
- Application server class loaders persist for the duration of the application server.
- Application class loaders persist until an application is stopped or dynamically reloaded.

  If a shared library that is configured with a native library path is associated with an application, whenever the application is restarted or dynamically reloaded the application might fail with an UnsatisfiedLinkError indicating that the library is already loaded. The error occurs because, when the application restarts, it invokes the shared library class to reload the native library. The native library, however, is still loaded in memory because the application class loader which previously loaded the native library has not yet been garbage collected.
- Only the JVM class loader can load a dependent native library.

  For example, if *NativeLib1* is dependent on *NativeLib2*, then *NativeLib2* must be visible to the JVM class loader. The path containing *NativeLib2* must be specified on Java library path defined by the LIBPATH environment variable. If a native library configured in a shared library is dependent on other native libraries, the dependent libraries must be configured on the LIBPATH of the JVM hosting the application server in order for that library to load successfully.

When configuring a shared library on a shared library settings page, if you specify a value for **Native Library Path**, the native libraries on this path are not located by the WebSphere Application Server application or shared library class loaders unless the class which loads the native library was itself loaded by the same class loader.

Because a native library cannot be loaded more than once by a class loader, it is preferable for native libraries to be loaded within shared libraries associated with the class loader of an application server, because these class loaders persist for the lifetime of the server.

1. Implement a static method in the class that loads the native library.

   In the class that loads the native library, call `System.loadLibrary(`*`native_library`*`)` in a static block. For example:

   ```
   static {System.loadLibrary("native_library");
   ```

   native_library loads during the static initialization of the class, which occurs exactly once when the class loads.
2. On the shared library settings page, set values for **Classpath** and **Native Library Path** that enable the shared library to load the native library.
3. Associate the shared library with the class loader of an application server.

   Associating a shared library with the class loader of an application server, rather than with an application, ensures that the shared library is loaded exactly once by the application server class loader, even though applications on the server are restarted or dynamically reloaded. Because the native library is loaded within a static block, the native library is never loaded more than once.

Shared library collection

Use this page to define a list of shared library files that deployed applications can use.

To view this administrative console page, click **Environment > Shared Libraries**.

Create a shared library for each library file that your application needs:

1. See what shared libraries are in a cell, node, or server.

   By default, a shared library is accessible to applications deployed (or installed) on the same node as the shared library file. Use the **Scope** field to change the scope to a different node or to a specific server. Under **Scope**, select the cell, a node, or a server and click **Apply**. Changing the scope of the collection table enables you to find shared libraries.

2. Select the scope for your new shared library.
3. Click **New**.
4. On the settings page for the new shared library, specify the name, class path, and any other variables for the library file that are needed.

After you create a shared library, associate it with an application or module or with the class loader of a server:

- To associate a shared library with an application or module, click **Applications > Enterprise Applications >** *application_name* **> Shared library references**. On the Shared library references page for the application, select the shared library file and click **OK**.
- To associate a shared library with a server class loader, click **Servers > Application servers >** *server_name* **> Java and Process Management > Class loader >** *class_loader_ID* **> Shared library references >** *shared_library_name*. On the settings page for the library reference for the server class loader, specify values that identify the shared library file.

Name

Specifies a name for the shared library.

Description

Describes the shared library file.

Shared library settings

Use this page to make a library file available to deployed applications.

To view this administrative console page, click **Environment > Shared Libraries >** *shared_library_name*.

Scope:

Specifies whether the shared library has its configuration file in a location that pertains to the cell, node, server, or cluster level.

Data type	String

Name:

Specifies a name for the shared library.

Data type	String

Description:

Describes the shared library file.

Data type	String

Classpath:

Specifies the class path used to locate the JAR files for the shared library support.

Data type	String
Units	Class path

Native Library Path:

Specifies the class path for locating platform-specific library files for shared library support; for example, `.dll`, `.so`, or *SRVPGM objects.

If you specify a value for **Native Library Path**, the native libraries are not located by the WebSphere Application Server application or shared library class loaders unless the following conditions exist:

- A class loads the native libraries.
- The application invokes a method in this class which loads the libraries.

  For example, in the class that loads the native library, call `System.loadLibrary(`*`native_library`*`)` in a static block:

  ```
  static {System.loadLibrary("native_library");
  ```
- The **Classpath** specified on this page contains the class that loads the libraries.

Native libraries cannot be loaded more than once by a class loader. Thus, it is preferable for native libraries to be loaded within shared libraries associated with the class loader of an application server.

Data type	String
Units	Class path

Associating shared libraries with applications or modules

You can associate a shared library with an application or module. Classes represented by the shared library are then loaded in the application's class loader, making the classes available to the application.

This topic assumes that you have defined a shared library at the cell, node, server, or cluster level. The shared library represents a library file used by multiple deployed applications.

This topic also assumes that you want to use the administrative console, and not an installed optional package, to associate a shared library with an application.

To associate a shared library with an application or module, create and configure a library reference using the administrative console. A library reference specifies the name of the shared library file.

If you associate a shared library with an application, do not associate the same shared library with a server class loader.

1. Click **Applications** > **Enterprise Applications** > *application_name* > **Shared library references** in the console navigation tree to access the Shared library references page.
2. On the Shared library references page, select an application or module to which you want to associate a shared library.
3. Click **Reference shared libraries**.
4. On the Shared Library Mapping page, select one or more shared libraries that the application or modules uses in the **Available** list, click >> to add them to the **Selected** list, and click **OK**.
5. Repeat steps 2 through 4 until you define a library reference instance for each shared library that your application or module requires.
6. On the Shared library references page, click **OK**.

When you run the application, classes represented by the shared library are loaded in the application's class loader, making the classes available to the application or module.

Shared library reference and mapping settings

Use the Shared library references and Shared Library Mapping pages to associate defined shared libraries with an application or Web module. A shared library is an external Java archive (JAR) file that is used by one or more applications. Using shared libraries enables multiple applications deployed on a server to use a single library, rather than use multiple copies of the same library. After you associate shared libraries with an application or module, the application or module class loader loads classes represented by the shared libraries and makes those classes available to the application or module.

To view the Shared library references console page, click **Applications > Enterprise Applications >** *application_name* **> Shared library references**. To view the Shared Library Mapping page, click **Reference shared libraries** on the Shared library references page. These pages are the same as the Shared Library Mapping for Modules and Shared Library Mapping pages in the application installation and update wizards.

On the Shared library references page, the first element listed is the application. The other elements are modules in the application.

To associate shared libraries with your application or module:

1. Select an application or module.
2. Click **Reference shared libraries**.
3. On the Shared Library Mapping page, select one or more shared libraries that the application or modules uses in the **Available** list, click **>>** to add them to the **Selected** list, and click **OK**.

A defined shared library for a file that your application or module uses must exist to associate your application or module to the library.

If no shared libraries are defined and the application is installed already, on the Shared Library Mapping page, click **New** and define a shared library.

You can otherwise define a shared library as follows:

1. Click **Environment > Shared Libraries**.
2. Specify whether the shared library is visible at the cell, node or server level.
3. Click **New**.
4. On the settings page for the new shared library, specify a name and one or more class paths. If the libraries are platform-specific files such as `.dll`, `.so`, or *SRVPGM objects, also specify a native library path. Then, click **Apply**.
5. Save the administrative configuration.

Application:

Specifies the name of the application that you are installing or that you selected on the Enterprise Applications page.

Module:

Specifies the name of the module associated with the shared libraries.

URI:

Specifies the location of the module relative to the root of the application EAR file.

Shared libraries:

Specifies the name of the shared library files associated with the application or module.

Associating shared libraries with servers

You can associate shared libraries with the class loader of a server. Classes represented by the shared library are then loaded in a server-wide class loader, making the classes available to all applications deployed on the server.

This topic assumes that you have defined a shared library at the cell, node, server, or cluster level. The shared library represents a library file used by multiple deployed applications.

To associate a shared library with the class loader of a server, create and configure a library reference using the administrative console. A library reference specifies the name of the shared library file.

If you associate a shared library with a server class loader, do not associate the same shared library with an application.

1. Configure class loaders for applications deployed on the server.
   a. Click **Servers > Application Servers >** ***server_name*** to access the settings page for the application server.
   b. Set values for the application **Class loader policy** and **Class loading mode** of the server.

      For information on these settings, see “Application server settings” on page 255 and the *Administering applications and their environment* PDF.
2. Create a library reference for each shared library file that your application needs.
   a. In the administrative console, click **Servers > Application servers >** *server_name* **> Java and Process Management > Class loader >** *class_loader_ID*.
   b. Click **Shared library references** to access the Library Reference page.
   c. Click **Add**.
   d. On the settings page for a library reference, name the library reference. The name identifies the shared library file that your application uses.
   e. Click **Apply**. The name of the library reference is shown in the list on the Library Reference page.

   Repeat the previous steps until you define a library reference for each shared library that your application needs.

Installed optional packages

Installed optional packages enable applications to use the classes in Java archive (`.jar`) files without having to include them explicitly in a class path. An installed optional package is a `.jar` file containing specialized tags in its manifest file that enable the application server to identify it. An installed optional package declares one or more shared library `.jar` files in the manifest file of an application. When the application is installed on a server or cluster, the classes represented by the shared libraries are loaded in the class loader of the application, making the classes available to the application.

When a Java 2 Platform, Enterprise Edition (J2EE) application is installed on a server or cluster, dependency information is specified in its manifest file. WebSphere Application Server reads the dependency information of the application (`.ear` file) to automatically associate the application with an installed optional package `.jar` file. WebSphere Application Server adds the `.jar` files in associated optional packages to the application class path. Classes in the installed optional packages are then available to application classes.

Installed optional packages used by WebSphere Application Server are described in section 8.2 of the J2EE specification, Version 1.4 at http://java.sun.com/j2ee/j2ee-1_4-fr-spec.pdf.

WebSphere Application Server supports using the manifest file (`manifest.mf`) in shared library `.jar` files and application `.ear` files. WebSphere Application Server does not support the Java 2 Platform Standard Edition (J2SE) Installed Optional Package semantics used in the J2SE specification (http://java.sun.com/j2se/1.3/docs/guide/extensions/spec.html), which primarily serve the applet environment. WebSphere Application Server ignores applet-specific tags within manifest files.

Sample manifest.mf file

A sample manifest file follows for an application app1.ear that refers to a single shared library file util.jar:

```
app1.ear:
    META-INF/application.xml
    ejb1.jar:
         META-INF/MANIFEST.MF:
             Extension-List: util
             util-Extension-Name: com/example/util
             util-Specification-Version: 1.4
         META-INF/ejb-jar.xml

util.jar:
    META-INF/MANIFEST.MF:
        Extension-Name: com/example/util
        Specification-Title: example.com's util package
        Specification-Version: 1.4
        Specification-Vendor: example.com
        Implementation-Version: build96
```

The syntax of a manifest entry depends on whether the entry applies to a member with a defining role (the shared library) or a member with a referencing role (a J2EE application or a module within a J2EE application).

Manifest entry tagging

Main tags used for manifest entries include the following:

Extension-List
A required tag with variable syntax. Within the context of the referencing role (application's manifest), this is a space delimited list that identifies and constructs unique Extension-Name, Extension-Specification tags for each element in the list. Within the context of the defining role (shared library), this tag is not valid.

Extension-Name
A required tag that provides a name and links the defining and referencing members. The syntax of the element within the referencing role is to prefix the element with the <ListElement> string. For each element in the Extension-List, there is a corresponding <ListElement>-Extension-Name tag. The defining string literal value for this tag (in the above sample com/example/util1) is used to match (in an equality test) the corresponding tags between the defining and referencing roles.

Specification-Version
A required tag that identifies the specification version and links the defining and referencing members.

Implementation-Version
An optional tag that identifies the implementation version and links the defining and referencing members.

Further information on these tags is in the .jar file specification at http://java.sun.com/j2se/1.4.2/docs/guide/jar/jar.html#Manifest%20Specification.

Using installed optional packages

You can associate one or more shared libraries with an application using an installed optional package that declares the shared libraries in the application's manifest file. Classes represented by the shared libraries are then loaded in the application's class loader, making the classes available to the application.

Read about installed optional packages in “Installed optional packages” on page 232 and in section 8.2 of the Java 2 Platform, Enterprise Edition (J2EE) specification, Version 1.4 at http://java.sun.com/j2ee/j2ee-1_4-fr-spec.pdf.

WebSphere Application Server does not support the Java 2 Platform Standard Edition (J2SE) Installed Optional Package semantics used in the J2SE specification (http://java.sun.com/j2se/1.3/docs/guide/extensions/spec.html), which primarily serve the applet environment. WebSphere Application Server ignores applet-specific tags within manifest files.

Installed optional packages expand the existing shared library capabilities of an application server. Prior to Version 6, an administrator was required to associate a shared library to an application or server. Installed optional packages enable an administrator to declare a dependency in an application’s manifest file to a shared library, with installed optional package elements listed in the manifest file, and automatically associate the application to the shared library. During application installation, the shared library `.jar` file is added to the class path of the application class loader.

If you use an installed optional package to associate a shared library with an application, do not associate the same shared library with an application class loader or a server class loader using the administrative console.

1. Assemble the library file, including the manifest information that identifies it as an extension. Two sample manifest files follow. The first sample manifest file has application `app1.ear` refer to a single shared library file `util.jar`:

   ```
   app1.ear:
       META-INF/application.xml
       ejb1.jar:
            META-INF/MANIFEST.MF:
                Extension-List: util
                util-Extension-Name: com/example/util
                util-Specification-Version: 1.4
            META-INF/ejb-jar.xml

   util.jar:
       META-INF/MANIFEST.MF:
           Extension-Name: com/example/util
           Specification-Title: example.com's util package
           Specification-Version: 1.4
           Specification-Vendor: example.com
           Implementation-Version: build96
   ```

   The second sample manifest file has application `app1.ear` refer to multiple shared library `.jar` files:

   ```
   app1.ear:
       META-INF/application.xml
       ejb1.jar:
            META-INF/MANIFEST.MF:
                Extension-List: util1 util2 util3
                Util1-Extension-Name: com/example/util1
                Util1-Specification-Version: 1.4
                Util2-Extension-Name: com/example/util2
                Util2-Specification-Version: 1.4
                Util3-Extension-Name: com/example/util3
                Util3-Specification-Version: 1.4
            META-INF/ejb-jar.xml

   util1.jar:
       META-INF/MANIFEST.MF:
           Extension-Name: com/example/util1
           Specification-Title: example.com's util package
           Specification-Version: 1.4
           Specification-Vendor: example.com
           Implementation-Version: build96
   ```

```
util2.jar:
    META-INF/MANIFEST.MF:
        Extension-Name: com/example/util2
        Specification-Title: example.com's util package
        Specification-Version: 1.4
        Specification-Vendor: example.com
        Implementation-Version: build96

util3.jar:
    META-INF/MANIFEST.MF:
        Extension-Name: com/example/util3
        Specification-Title: example.com's util package
        Specification-Version: 1.4
        Specification-Vendor: example.com
        Implementation-Version: build96
```

2. Create a shared library that represents the library file assembled in step 1. This installs the library file as a WebSphere Application Server shared library.
3. Copy the shared library `.jar` file to the cluster members.
4. Assemble the application, declaring in the application manifest file dependencies to the library files named the manifest created for step 1.

   See the *Developing and deploying applications* PDF for more information.
5. Install the application on the server or cluster.

   See the *Developing and deploying applications* PDF for more information.

During application installation, the shared library `.jar` files are added to the class path of the application class loader.

Library reference collection

Use this page to view and manage library references that define how to use global libraries. For example, you can use this page to associate shared library files with a deployed application.

To view this administrative console page, click **Servers > Application servers >** *server_name* **> Java and Process Management > Class loader >** *class_loader_ID* **> Shared library references**.

If no shared libraries are defined in your environment, such as at the node or server scope, after you click **Add** a message is displayed stating that you must define a shared library before you can create a library reference. A shared library is a container-wide library file that can be used by deployed applications. To define a shared library, click **Environment > Shared Libraries** and specify the scope of the container. Then, click **New** and specify a name and one or more paths for the shared library. After you define a shared library, return to this page, click **Add**, and create a library reference.

Library name

Specifies a name for the library reference.

Library reference settings

Use this page to define library references, which specify how to use global libraries.

To view this administrative console page, click **Servers > Application servers >** *server_name* **> Java and Process Management > Class loader >** *class_loader_ID* **> Shared library references >** *library_reference_name*.

A shared library is a container-wide library file that can be used by deployed applications. To define a shared library, click **Environment > Shared Libraries** and specify the scope of the container. Then, click **New** and specify a name and one or more paths for the shared library.

Library name:

Specifies the name of the shared library to use for the library reference.

Data type	String

Environment: Resources for learning

Use the following links to find relevant supplemental information about configuring the WebSphere Application Server environment. 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.

Programming instructions and examples

- WebSphere Application Server education

Administration

- Listing of all IBM WebSphere Application Server Redbooks

Chapter 6. Working with server configuration files

This topic show how to manage application server configuration files.

Application server configuration files define the available application servers, their configurations, and their contents.

You should periodically save changes to your administrative configuration. You can change the default locations of configuration files, as needed.

- Edit configuration files.

  The master repository is comprised of .xml configuration files

  You can edit configuration files using

  – The administrative console. See the Using the administrative console topic in the *Using the administrative clients* PDF.

  – Scripting. See the Getting started with scripting topic in the *Using the administrative clients* PDF.

  – The wsadmin commands. See the Using command line tools topic in the *Using the administrative clients* PDF.

  – Programing. See the Using administrative programs (JMX) topic in the *Using the administrative clients* PDF.

  – By editing a configuration file directly.

  The configuration files are in ASCII format. You cannot edit them in the Hierarchical File System (HFS). Instead, transfer the files to your workstation, edit them, and then transfer them back to the HFS.
- Save changes made to configuration files. Using the console, you can save changes as follows:
  1. In the navigation select **System Administration > Save changes to master repository**.
  2. Put a check mark in the **Synchronize changes with Nodes** check box.
  3. Click **Save**.
- Handle temporary configuration files resulting from a session timing out.
- Change the location of temporary configuration files.
- Change the location of backed-up configuration files.
- Change the location of temporary workspace files.
- Back up and restore configurations.

Configuration documents

WebSphere Application Server stores configuration data for servers in several documents in a cascading hierarchy of directories. The configuration documents describe the available application servers, their configurations, and their contents. Most configuration documents have XML content.

Hierarchy of directories of documents

The cascading hierarchy of directories and the documents' structure support multinode replication to synchronize the activities of all servers in a cell. In a Network Deployment environment, changes made to configuration documents in the cell repository, are automatically replicated to the same configuration documents that are stored on nodes throughout the cell.

At the top of the hierarchy is the **cells** directory. It holds a subdirectory for each cell. The names of the cell subdirectories match the names of the cells. For example, a cell named *cell1* has its configuration documents in the subdirectory *cell1*. The name of the cell must be different from the cluster name pair.

On the Network Deployment node, the subdirectories under the cell contain the entire set of documents for every node and server throughout the cell. On other nodes, the set of documents is limited to what applies to that specific node. If a configuration document only applies to *node1*, then that document exists in the configuration on *node1* and in the Network Deployment configuration, but not on any other node in the cell.

Each cell subdirectory has the following files and subdirectories:

- The `cell.xml` file, which provides configuration data for the cell
- Files such as `security.xml`, `virtualhosts.xml`, `resources.xml`, and `variables.xml`, which provide configuration data that applies across every node in the cell
- The **clusters** subdirectory, which holds a subdirectory for each cluster defined in the cell. The names of the subdirectories under clusters match the names of the clusters.

  Each cluster subdirectory holds a cluster.xml file, which provides configuration data specifically for that cluster.
- The **nodes** subdirectory, which holds a subdirectory for each node in the cell. The names of the nodes subdirectories match the names of the nodes.

  Each node subdirectory holds files such as `variables.xml` and `resources.xml`, which provide configuration data that applies across the node. Note that these files have the same name as those in the containing cell's directory. The configurations specified in these node documents override the configurations specified in cell documents having the same name. For example, if a particular variable is in both cell- and node-level `variables.xml` files, all servers on the node use the variable definition in the node document and ignore the definition in the cell document.

  Each node subdirectory holds a subdirectory for each server defined on the node. The names of the subdirectories match the names of the servers. Each server subdirectory holds a `server.xml` file, which provides configuration data specific to that server. Server subdirectories might hold files such as `security.xml`, `resources.xml` and `variables.xml`, which provide configuration data that applies only to the server. The configurations specified in these server documents override the configurations specified in containing cell and node documents having the same name.
- The **applications** subdirectory, which holds a subdirectory for each application deployed in the cell. The names of the applications subdirectories match the names of the deployed applications.

  Each deployed application subdirectory holds a `deployment.xml` file that contains configuration data on the application deployment. Each subdirectory also holds a **META-INF** subdirectory that holds a J2EE application deployment descriptor file as well as IBM deployment extensions files and bindings files. Deployed application subdirectories also hold subdirectories for all .war and entity bean .jar files in the application. Binary files such as .jar files are also part of the configuration structure.

An example file structure is as follows:

```
cells
  cell1
     cell.xml resources.xml virtualhosts.xml variables.xml security.xml
     nodes
        nodeX
           node.xml variables.xml resources.xml serverindex.xml
           serverA
              server.xml variables.xml
           nodeAgent
              server.xml variables.xml
        nodeY
           node.xml variables.xml resources.xml serverindex.xml
     applications
        sampleApp1
           deployment.xml
           META-INF
              application.xml ibm-application-ext.xml ibm-application-bnd.xml
        sampleApp2
           deployment.xml
           META-INF
              application.xml ibm-application-ext.xml ibm-application-bnd.xml
```

Changing configuration documents

You can use one of the administrative tools (console, wsadmin, Java APIs) to modify configuration documents or edit them directly. It is preferable to use the administrative console because it validates changes made to configurations. "“Configuration document descriptions”" states whether you can edit a document using the administrative tools or must edit it directly.

Configuration document descriptions

Most configuration documents have XML content. The table describes the documents and states whether you can edit them using an administrative tool or must edit them directly.

If possible, edit a configuration document using the administrative console because it validates any changes that you make to configurations. You can also use one of the other administrative tools (wsadmin or Java APIs) to modify configuration documents. Using the administrative console or wsadmin scripting to update configurations is less error prone and likely quicker and easier than other methods.

However, you cannot edit some files using the administrative tools. Configuration files that you must edit manually have an X in the **Manual editing required** column in the table below.

Document descriptions

(The paths in the Locations column are split on multiple lines for publishing purposes.)

Configuration file	Locations	Purpose	Manual editing required
`admin-authz.xml`	`config/cells/` *`cell_name/`*	Define a role for administrative operation authorization.	X

`app.policy`	`config/cells/`
	`cell_name/`
	`nodes/`*`node_name/`*
		Define security permissions for application code.	X

`cell.xml`	`config/cells/` *`cell_name/`*	Identify a cell.	

`deployment.xml`	`config/cells/`
	`cell_name/`
	`applications/`
	`application_name/`
		Configure application deployment settings such as target servers and application-specific server configuration.	

`filter.policy`	`config/cells/` *`cell_name/`*	Specify security permissions to be filtered out of other policy files.	X
`integral-jms-authorizations.xml`	`config/cells/` *`cell_name/`*	Provide security configuration data for the integrated messaging system.	X

`library.policy`	`config/cells/`
	`cell_name/`
	`nodes/`*`node_name/`*
		Define security permissions for shared library code.	X

`namestore.xml`	`config/cells/` *`cell_name/`*	Provide persistent name binding data.	X
`naming-authz.xml`	`config/cells/` *`cell_name/`*	Define roles for a naming operation authorization.	X

`node.xml`	`config/cells/`
	`cell_name/`
	`nodes/`*`node_name/`*
		Identify a node.	

resources.xml	config/cells/*cell_name*/ config/cells/*cell_name*/nodes/*node_name*/ config/cells/*cell_name*/nodes/*node_name*/servers/*server_name*/	Define operating environment resources, including JDBC, JMS, JavaMail, URL, JCA resource providers and factories.	
security.xml	config/cells/*cell_name*/	Configure security, including all user ID and password data.	
server.xml	config/cells/*cell_name*/nodes/*node_name*/servers/*server_name*/	Identify a server and its components.	
serverindex.xml	config/cells/*cell_name*/nodes/*node_name*/	Specify communication ports used on a specific node.	
spi.policy	config/cells/*cell_name*/nodes/*node_name*/	Define security permissions for service provider libraries such as resource providers.	X
variables.xml	config/cells/*cell_name*/ config/cells/*cell_name*/nodes/*node_name*/ config/cells/*cell_name*/nodes/*node_name*/servers/*server_name*/	Configure variables used to parameterize any part of the configuration settings.	
virtualhosts.xml	config/cells/*cell_name*/	Configure a virtual host and its MIME types.	

Object names: What the name string cannot contain

When you create a new object using the administrative console or a wsadmin command, you often must specify a string for a name attribute.

Most characters are allowed in the name string. However, the name string cannot contain the following characters. The name string also cannot contain leading and trailing spaces.

/	forward slash
\	backslash
*****	asterisk
,	comma

:	colon
;	semi-colon
=	equal sign
+	plus sign
?	question mark
|	vertical bar
<	left angle bracket
>	right angle bracket
&	ampersand (and sign)
%	percent sign
’	single quote mark
″	double quote mark
]]>	No specific name exists for this character combination.
.	period (not valid if first character; valid if a later character)
#	Hash mark
$	Dollar sign

Configuration repositories

A configuration repository stores configuration data.

By default, configuration repositories reside in the *config* subdirectory of the product installation root directory.

A cell-level repository stores configuration data for the entire cell and is managed by a file repository service that runs in the deployment manager. The deployment manager and each node have their own repositories. A node-level repository stores configuration data that is needed by processes on that node and is accessed by the node agent and application servers on that node.

When you change a WebSphere Application Server configuration by creating an application server, installing an application, changing a variable definition or the like, and then save the changes, the cell-level repository is updated. The file synchronization service distributes the changes to the appropriate nodes.

Handling temporary configuration files resulting from session timeout

If the console is not used for 15 minutes or more, the session times out. The same thing happens if you close the browser window without saving the configuration file. Changes to the file are saved to a temporary file when the session times out, after 15 minutes. This topic discusses what happens depending on whether you load the saved file.

A configuration file must have been saved from a previous administrative console session for the user ID that you are currently using to access the administrative console.

When a session times out, the configuration file in use is saved under the `userid/timeout` directory under the ServletContext’s temp area. This value is the value of the javax.servlet.context.tempdir attribute of the ServletContext context. By default, it is: `*profile_root*/temp/hostname/Administration/admin/admin.war`

You can change the temp area by specifying it as a value for the tempDir init-param of the action servlet in the deployment descriptor (web.xml) of the administrative application.

The configuration file is also saved automatically when the same user ID logs into the non-secured console again, effectively starting a different session. This process is equivalent to forcing the existing user ID out of session, similar to a session timing out.

The next time you log on to the administrative console, you are prompted to load the saved configuration file. Do one of the following actions:

- Load the saved file.
  1. If a file with the same name exists in the *profile_root*/config directory, that file is moved to the userid/backup directory in the temp area.
  2. The saved file is moved to the *profile_root*/config directory.
  3. The file is then loaded.
- Do not load the saved file.

  The saved file is deleted from the userid/timeout directory in the temp area.

You loaded the saved configuration file if you chose to do so.

Once you have logged into the administrative console, do whatever administration of WebSphere Application Server that you need to do.

Changing the location of temporary configuration files

You can change the default directory where temporary configuration files are stored.

The configuration repository uses copies of configuration files and temporary files while processing repository requests. It also uses a backup directory while managing the configuration. You can change the default locations of these files from the configuration directory to a directory of your choice by using the administrative console.

The default location for the configuration temporary directory is *profile root*/config/temp. Change the location by doing the following actions:

1. Click **Servers > Application servers** in the navigation tree of the administrative console. Then, click ***server name*** **> Administration > Administration services > Repository service > Custom properties**.
2. On the Properties page, click **New**.
3. On the settings page for a property, define a property for the temporary file location. The key for this property is was.repository.temp. The value is the full path name to the desired location.
4. Click **OK**.

Changing the location of backed-up configuration files

You can change the default directory where backup files are stored.

During administrative processes like adding a node to a cell or updating a file, configuration files are temporarily backed up to a backup location.

The default location for the backup configuration directory is *profile root*/config/backup. Change the location by doing the following actions:

1. Click **Servers > Application servers** in the navigation tree of the administrative console. Then, click ***server name*** **> Administration > Administration services > Repository service > Custom properties**.
2. On the Properties page, click **New**.
3. On the settings page for a property, define a property for the backup file location. The key for this property is was.repository.backup. The value is the full path name to the desired location.
4. Click **OK**.

Changing the location of temporary workspace files

The default workspace root is calculated based on the user installation root. This topic discusses how to change the location of temporary workspace files.

You must first install WebSphere Application Server before you change the location of temporary workspace files.

With the administrative console workspace, client applications can navigate the configuration. Each workspace has its own repository location defined either in the system property or the property that is passed to a workspace manager when creating the workspace: workspace.user.root or workspace.root, which is calculated as %workspace.root%/*user_ID*/workspace/wstemp.

The default workspace root is calculated based on the user installation root: %user.install.root%/wstemp. You can change the default location of temporary workspace files:

Change the setting for the Java system property *workspace.user.root* or *workspace.root* so its value is no longer set to the default location.

Set the Java system property in the servant process when launching a Java process using the -D option. For example, to set the default location to the full path of the root of all users' directories, use the following option:

```
-Dworkspace.user.root=full_path_for_root_of_all_user_directories
```

You changed the location of temporary workspace files.

Backing up and restoring administrative configuration files

This topic discusses how to back up and restore administrative configuration files.

WebSphere Application Server represents its administrative configurations as XML files. You should back up configuration files on a regular basis.

Restore the configuration only if the configuration files that you backed up are at the same level of the release, including fixes, as the release to which you are restoring.

1. Synchronize administrative configuration files.
   a. Click **System Administration > Nodes** in the console navigation tree to access the Nodes page.
   b. Click **Full Resynchronize**. The resynchronize operation resolves conflicts among configuration files and can take several minutes to run.
2. Run the backupConfig command to back up configuration files. See the backupConfig command topic in the *Using the administrative clients* PDF for information.
3. Run the restoreConfig command to restore configuration files. See the restoreConfig command topic in the *Using the administrative clients* PDF for information. Specify backup files that do not contain invalid or inconsistent configurations.

Transformation of configuration files

The WebSphere Application Server master configuration repository stores configuration files for all the nodes in the cell. When you upgrade the deployment manager from one release of WebSphere Application Server to another, the configuration files that are stored in the master repository for the nodes on the old release are converted into the format of the new release.

With this conversion, the deployment manager can process the configuration files uniformly. However, nodes on an old release cannot readily use configuration files that are in the format of the new release.

WebSphere Application Server addresses the problem when it synchronizes the configuration files from the master repository to a node on an old release. The configuration files are first transformed into the old release format before they ship to the node. WebSphere Application Server performs the following transformations on configuration documents:

- Changes the XML name space from the format of the new release to the format of the old release
- Strips out attributes of cell-level documents that are applicable to the new release only
- Strips out new resource definitions that are not understood by old release nodes

Backing up the WebSphere Application Server for z/OS system

This topic discusses methods for backing up configuration and data for the WebSphere Application Server for z/OS system.

Use the following guidelines to back up parts of your WebSphere Application Server for z/OS system.

1. **Back up the HFS that contains your WebSphere Application Server for z/OS configuration** (e.g. WebSphere/V5R0M0/AppServer).
2. Back up the RMDATA log for Resource Recovery Service (RRS). Otherwise, a failure could force you to do a cold start of RRS.
3. Set the ARCHIVE log retention period to one day.
4. Incorporate the following items in your normal backup procedures:
   - WebSphere Application Server for z/OS procedure libraries.
   - WebSphere Application Server for z/OS load libraries.
   - The directory where WebSphere Application Server for z/OS run-time information is written. The default is */WebSphere/V5R0M0*.
5. Back up your own application executable files, databases, and bindings.
6. If you wish to back up a single server, you can use the export/import function in the Administrative Console. For details on how to do this, see the assembling applications information.

Server configuration files: Resources for learning

Use the following links to find relevant supplemental information about administering WebSphere Application Server configuration files. 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:

Administration

- IBM WebSphere Application Server Redbooks

  This site contains a listing of all WebSphere Application Server Redbooks.
- IBM WebSphere developerWorks

  This site is the home of technical information for developers working with WebSphere products. You can download WebSphere software, take a fast path to developerWorks zones, such as VisualAge Java or WebSphere Application Server, learn about WebSphere products through a newcomers page, tutorials, technology previews, training, and Redbooks, get answers to questions about WebSphere products, and join the WebSphere community, where you can keep up with the latest developments and technical papers.
- WebSphere Application Server Support page

Take advantage of the Web-based Support and Service resources from IBM to quickly find answers to your technical questions. You can easily access this extensive Web-based support through the IBM Software Support portal at URL `http://www-3.ibm.com/software/support/` and search by product category, or by product name. For example, if you are experiencing problems specific to WebSphere Application Server, click **WebSphere Application Server** in the product list. The WebSphere Application Server Support page appears.

Chapter 7. Administering application servers

An application server configuration provides settings that control how an application server provides services for running applications and their components.

Install WebSphere Application Server.

After you install WebSphere Application Server you might have to perform some of the following tasks. Other then creating application servers, these tasks can be performed in any order.

- Create application servers.
- Create clusters.
- Configure transport chains.
- Set up peer restart and recovery
- Develop custom services.
- Define processes for the application server.

  As part of defining processes, you can define:

  1. Define process execution statements for starting or initializing a UNIX process.
  2. Configure monitoring policies to track the performance of a process.
  3. Configure the process logs to which standard out and standard error streams write.
  4. Configure name-value pairs for properties.
- Configure the Java virtual machine.

After preparing a server, deploy an application or component on the server. See the *Administering applications and their environment* PDF for a sample procedure that you might follow in configuring the application server runtime and resources.

Manage the application servers you created.

Application servers

Application servers extend the ability of a Web server to handle Web application requests, typically using Java technology. An application server makes it possible for a server to generate a dynamic, customized response to a client request.

For example, given:

1. A user at a Web browser on the Internet visits a company Web site. The user requests to use an application that provides access to data in a database.
2. The user request flows to the Web server.
3. The Web server determines that the request involves an application containing resources not handled directly by the Web server (such as servlets). It forwards the request to WebSphere Application Server.
4. WebSphere Application Server forwards the request to one of its application servers on which the application is running.
5. The invoked application then processes the user request. For example:
   - An application servlet prepares the user request for processing by an enterprise bean that performs the database access.
   - The application produces a dynamic Web page containing the results of the user query.
6. The application server collaborates with the Web server to return the results to the user at the Web browser.

WebSphere Application Server provides multiple application servers that can be either separately configured processes or nearly identical clones.

Application server naming conventions

Before you install a new WebSphere Application Server for z/OS environment, it is important to carefully plan your naming convention. Your naming convention should be able to grow with your system when you increase the number of cells, nodes, servers, and clusters. It should also be able to accommodate Sysplex and LPAR names, as well as instances such as test, integration, and production stages in your environment.

WebSphere Application Server for z/OS servers are like IMS or CICS regions.

- They contain tailored procedures for the controllers and servants.
- They contain tailored environmental variables for each instance of a server.
- They use WLM Classification of regions, working within the regions, and are defined as application environments.
- They may be self-contained or dependent on other servers.
- They need RACF definitions for Control and Server STC (user IDs, resource profiles), as well as UNIX permissions.
- Their users must be allowed to access the servers and to use various objects within them.

A WebSphere Application Server for z/OS environment consists of a number of address spaces which require the installation to manage security profiles, workload classification constructs, and so on. To create, manage, and recognize application servers, it may be helpful to create a template for naming your servers and server instances. You can find an example template in the *Installing your application serving environment* PDF.

It is also important to plan the naming conventions for your data sets carefully.

- SMP/E target data sets, depending on your maintenance process (regular data sets and the HFS, including its mount points)
- Customization HFS, including its mount point
- HLQ for your customization data sets (*.CNTL, *.DATA, and *.SAVDCFG)
- Error logstream names
- DB2 collection and package names

Sample configuration and naming conventions

This article offers a simple configuration template and examples for naming.

This configuration is an alternate example, provided by a customer, of what you can do with your configuration.

Scenario

Data sets and HFS

HLQ code libraries (PDSE data sets)

BBO51.* (BBO51.SBBOLOAD for example)

z/OS data set names used for the customization

```
High level qualifier for customization data sets:
          <id>.<sysplex/sysname>.V5CELLA.<step>
          <id>                  = your preferred first HLQ
          <sysplex/sysname>  = location identifier
```

<step>	
SECDOM	Configure Security Domain (*.CNTL, *.DATA, *.SAVECFG)
GENERAL	Data set for general customzation parameters (*.SAVECFG)
BAPSRV01	Configure base application server node 01 (*.CNTL, *.DATA)
BAPSRV02	Configure base application server node 02 (*.CNTL, *.DATA)
JMS	Configure integral JMS provider (*.CNTL, *.DATA)
DMGR	Configure deployment manager node (*.CNTL, *.DATA)
BAPSRV01.FEDERAT	Federate base application server node (*.CNTL, *.DATA)

HFS Data sets

Base application server Nodes:

OMVS.<sysname>.V5CELLA.BAPSRV01.CONFIG.HFS
OMVS.<sysname>.V5CELLA.BAPSRV02.CONFIG.HFS

Deployment Manager:

OMVS.<sysplex>.V5CELLA.DMGR.CONFIG.HFS

Mount points Installation HFS:

WebSphere Application Server SMP/E home directory:
/usr/lpp/zWebSphere/V5R1M0

WebSphere JMS Client Java™ Feature SMP/E home directory:
/usr/lpp/mqm/V5R3M1

Java home directory:
/usr/lpp/java/J1.4

Mount point Base application server:

/WebSphere/V5R1M0/v5cella/baps/bapsrv01
/WebSphere/V5R1M0/v5cella/baps/bapsrv01

Mount point Deployment Manager:

/WebSphere/V5R1M0/v5cella/dmgr

WebSphere Application Server home directory of the Deployment Manager:

/WebSphere/V5R1M0/v5cella/dmgr/dmgrcella

Location Service Daemon (Base application server) home directory:

/WebSphere/V5R1M0/v5cella/baps/bapsrv01/Daemon

Location Service Daemon (Deployment Manger) home directory:

/WebSphere/V5R1M0/v5cella/dmgr/Daemon

Naming Schema for the JCL Procedures of Cell A

Table 3.

V5ADC	Controller deployment manager
V5ADS	Servant deployment manager
V5ALC	Controller location service daemon
V5ASC	Controller application server and nodeagent
V5ASS	Servant application server
V5AAO	Asynchronous administration operation procedure
V5AWTR	Startprocedure CTRACE

Customization parameters

Procedures, jobnames, short and long names, RACF user and groups, USS permissions, and IP ports used.

SYSPLEX	TESTPLX1		Deployment Manager on LP01
Component	LPAR	Controller/ Server	″Short Names″
CELL A in TESTPLX1	LP01	BASE Cellname	V5ABA
		DMGR Cellname	V5A
Location Service Daemon #1 of Cell A	LP01	Controller	
		Procedurename	V5ALC
		BASE Jobname/ Node	V5A1LBA
		DMGR Jobname/ Node	V5A1LSD
Location Service Daemon #2 of Cell A	LP02	Controller	
		Procedurename	V5ALC
		BASE Jobname/ Node	V5A2LBA
		DMGR Jobname/ Node	V5A2LSD
DeploymentManager	LP01	Controller	
		Procedurename	V5ADC
		Jobname/Server	V5ADMGR
		Servant	
		Procedurename	V5ADS
		Jobname	V5ADMGRS
Nodeagent #1 of DeploymentManager	LP01	Controller	
		Procedurename	V5ASC
		Jobname/ Node	V5A1DNA
Nodeagent #2 of DeploymentManager	LP02	Controller	
		Procedurename	V5ASC
		Jobname/ Node	V5A2DNA

Server #1 in CELL A (belonging to Nodeagent #1)	LP01	Controller	
		Procedurename	V5ASC
		BASE Jobname/ Node	V5A1S01
		Servant	
		Procedurename	V5ASS
		BASE Jobname	V5A1S01S
		DMGR Jobname	V5A1S01S
Server #2 in CELL A (belonging to Nodeagent #2)	LP02	Controller	
		Procedurename	V5ASC
		BASE Jobname/ Node	V5A2S01
		Servant	
		Procedurename	V5ASS
		BASE Jobname	V5A2S01S
		DMGR Jobname	V5A2S01S
Admin asynch operation	LP01, LP02	Procedurename	V5AAO
Administrator ID	LP01, LP02		
Guest ID	LP01, LP02		

Creating application servers

WebSphere Application Server provides you with the capability to create application servers.

Determine if you want to use the application server you are creating as part of a cluster. If this application server is going to be part of a cluster, use the Create a new cluster wizard to create this application server.

You can use either the **createApplicationServer**, **createWebServer**, or the **createGenericServer** wsadmin commands (see the *Using the administrative clients* PDF), or the **Create New Application Server** wizard in the administrative console to create a new application server.

You can also create a new application server when you add a cluster member to a server cluster.

If you are migrating from a previous version of WebSphere Application Server, you can upgrade a portion of the nodes in a cell, while leaving others at the older product level. This means that, for a period of time, you might be managing servers that are running at two different release levels in the same cell. However, when you create a new server definition, you must use a server configuration template, and that template must be created from a server instance that matches the version of the node on which you are creating a server.

There are no restrictions on what you can do with the servers running on the newer release level.

To create a new application server:

1. In the administrative console, click **Servers > Application servers > New**. The Create New Application Server wizard starts.

   You can also create the new application server using the wsadmin **createApplicationServer** command. For a description of how to use this command, see the *Using the administrative clients* PDF.
2. Configure your application server.
   a. Select a node for the application server.
   b. Type in a name for the application server. The name must be unique within the node.

c. Click **Next**.

d. Select a server template for the new server. You can use a default application server template for your new server or you can use the template that is optimized to perform well for development uses. The new application server inherits all of the configuration settings of the template server.

e. Click **Next** and then select Generate Unique HTTP Ports if you want unique ports generated for the application server. By default, this option is enabled. If you select this option, you might need to update the alias list for the virtual host that you plan to use with this server to contain these new port values. If you deselect this option, ensure that the default port values do not conflict with other servers on the same physical machine.

f. Click **Next** and specify a short name for the server. The short name is also used as the JOBNAME for the server. The default value for a short name is BBOS*xxx*, where *xxx* is the first free number in the cell that can be used to create a unique short name. For example, if a default short name is already assigned to two other servers in the cell, the short name BBOS003 is assigned to this server if you do not specify a short name when you create this server.

g. Specify a generic short name for the server that is converted into the cluster short name if the server becomes a cluster member. The default value for a generic short name is BBOS*xxx*, where *xxx* is the first free number in the cell that can be used to create a unique short name. For example, if default generic short names are already assigned to three other servers in the cell, the generic short name BBOS004 is assigned to this server if you do not specify a generic short name when you create this server.

h. Click **Next**. Review the settings for the new server. If you want to change any of the settings, click **Previous** until you return to a page where you can change that setting. If you do not want to make any changes, click **Finish**

i. Click Review, select Synchronize changes with Nodes, and then click **Save** to save your changes.

The new application server appears in the list of servers on the administrative console Application Servers page.

This newly created application server is configured with many default settings that do not display when you run the Create New Application Server wizard. To view all of the configuration settings for this application server in the administrative console, click **Servers > Application servers** and then click on the name of this application server. If necessary, use this page to change the configuration settings for this server.

You can also Set the client.encoding.override JVM argument to UTF-8 if you need to use multiple language encoding support in the administrative console.

Configuring application servers for UCS Transformation Format

You can use the client.encoding.override=UTF-8 JVM argument to configure an application server for UCS Transformation Format. This format enables an application server to handle most character encodings, including specialized mathematical and technical symbols.

The client.encoding.override=UTF-8 argument is provided for backwards compatibility. You should only specify this argument if you require multiple language encoding support in the administrative console and there is no other way for you to set the request character encoding required to parse post and query strings.

Before configuring an application server for UCS Transformation Format, you should try to either:

- Explicitly set the ServletRequest Encoding inside of the JSP or Servlet that is receiving the POST and or query string data, which is the preferred J2EE solution, or
- Enable the autoRequestEncoding, option, which uses the client's browser settings to determine the appropriate character encoding. Older browsers might not support this option.

Important: If the client.encoding.override=UTF-8 JVM argument is specified, the autoRequestEncoding option does not work even if it is enabled. Therefore, when an application server receives a client request, it checks to see if the charset option is set on the content type header of the request:

1. If it is set, the application server uses the content type header for character encoding.
2. If it is not set, the application server uses the character encoding that is specified for the default.client.encoding system property.
3. If neither charset nor the default.client.encoding system property is set, the application server uses the ISO-8859-1 character set.

The application server never checks for an Accept-Language header. However, if the autoRequestEncoding option is working, the application server checks for an Accept-Language header before checking to see if a character encoding is specified for the default.client.encoding system property.

To configure an application server for UCS Transformation Format:

1. In the administrative console, click **Servers > Application servers** and select the server you want to enable for UCS Transformation Format.
2. Click **Java and Process Management > Process Definition > Java Virtual Machine**.
3. Specify `-Dclient.encoding.override=UTF-8` for **Generic JVM Arguments** and click **OK**. When this argument is specified, UCS Transformation Format is used instead of the character encoding that would be used if the autoRequestEncoding option was in effect.
4. Click **Save** to save your changes.
5. Restart the application server.

The application server uses UCS Transformation Format for encoding.

Managing application servers

You can use the administrative console or command line tools to manage your application servers.

You can use either the administrative console or command line tools to manage your application servers.

Important: If you are migrating from a previous version of WebSphere Application Server, you can upgrade a portion of the nodes in a cell, while leaving others at the older release level. This means that, for a period of time, you might be managing servers that are running at different release levels in the same cell. In this mixed environment, there are some restrictions on what you can do with servers that are running the older release level. There are no restrictions on what you can do with the servers that are running on the newer release level. For details, see “Creating application servers” on page 251 and “Creating clusters” on page 397.

To use the administrative console to view and manage an application server:

To use the administrative console to view and manage an application server:

1. In the administrative console click **Servers > Application servers**. The Application servers page lists the application servers in your environment and the status of each of these servers. You can use this page to change the status of a listed application server.
2. Click the name of a listed application server to view or change the configuration settings for that application server.
3. Save any configuration changes you make.
4. Start an application server.
5. Monitor the running application servers.
6. Stop an application server.

7. Delete an application server.

   Tip: If the server you are deleting has applications or modules mapped to it, remap the modules to another server, or create a new server and remap the modules to the new server, before deleting the old server. After a server to which modules are mapped is deleted, the modules cannot be remapped to another server. Therefore, if you do not remap the modules to another server before deleting the old one, you must uninstall all of the modules that were mapped to the old server, and then reinstall them on a different server.

   a. In the administrative console, click **Servers > Application servers** to access the Application Servers page.
   b. Select an application server to delete.
   c. Click **Delete**.
   d. Click **OK** to confirm the deletion.

The application servers are properly configured and the appropriate application servers are running.

Create additional application server as required.

Server collection

Use this page to view information about and manage application servers, Java message service (JMS) servers, and Web servers. For the Network Deployment product, you can also use this page to view information about and manage generic servers.

Application servers

The Application servers page lists the application servers in the cell. You can use this page to start and stop these application servers.

If you are using the Network Deployment product and have created clusters, you can also use this page to manage application servers that are cluster members if the **Include cluster members in the collection** console page preference is selected. When this preference is selected, a Cluster Name column is included in the application server information table. If an application server is part of a cluster, the Cluster Name column specifies the name of that cluster. If this preference is not selected, the Cluster Name column does not appear in the table and application servers that are cluster members are not listed in the list of application servers and cannot be managed from this page.

If you are using the Network Deployment product, this page also displays the status of the application servers. The status indicates whether a server is running, stopped, or encountering problems. You can also use this page to create new application servers, create application server templates, or delete existing application servers.

To view this administrative console page, click **Servers > Application servers**.

Generic servers

The Generic servers page is only available for the Network Deployment product. This page lists the generic servers in the cell and displays the status of these generic servers. The status indicates whether a server is running, stopped, or encountering problems. You can use this page to start and terminate these generic servers. You can also use this page to create new generic servers, create generic server templates, or delete existing generic servers. However, the Stop and Stop Immediate buttons on the administrative console do not work for generic servers.

To view this administrative console page, click **Servers > Generic servers**.

Java message service (JMS) servers

The JMS servers page lists the JMS servers in the cell. You can use this page to start and stop these JMS servers. Each JMS server provides the functions of the JMS provider for a node in your administrative domain. There can be at most one JMS server on each node in the administration domain, and any application server within the domain can access JMS resources served by any JMS server on any node in the domain. To view this administrative console page, click **Servers > JMS servers**.

Note: JMS servers apply only to WebSphere Application Server Version 5.x nodes. You cannot create a JMS server on a node that is running WebSphere Application Server 6.x, but the existing Version 5.x JMS servers continue to be displayed, and you can modify their properties. However, you cannot use this page to delete a Version 5.x JMS server.

Web servers

The Web servers page lists the Web servers in your administrative domain. You can use this page to generate and propagate a Web server plug-in configuration file, create new Web servers, create new Web server templates, or delete existing Web servers. You can also use this page to start and stop these Web servers. To view this administrative console page, click **Servers > Web servers**.

Name

Specifies a logical name for the server. For WebSphere Application Server, server names must be unique within a node.

For WebSphere Application Server for z/OS, this is sometimes called the long name. 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 uses the server name for administrative actions, such as referencing the server in scripting.

Node

Specifies the name of the node holding the server.

Version

Specifies the version of the WebSphere Application Server product on which the server runs.

Status

Indicates whether the server is started or stopped. (Network Deployment only)

Note that 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.

Application server settings

Use this page to view or change the settings of an application server instance. An application server is a server which provides services required to run enterprise applications.

To view this administrative console page, click **Servers > Application Servers >** *server_name*.

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 the server is running.

Name:

Specifies a logical name for the server. Server names must be unique within a node. However, for multiple nodes within a cluster, you may have different servers with the same server name as long as the server and node pair are unique. You cannot change the value that displays in this field.

For example, a server named *server1* in a node named *node1* in the same cluster with a server named *server1* in a node named *node2* is allowed. Configuring two servers named *server1* in the same node is not allowed. WebSphere Application Server uses the server name for administrative actions, such as referencing the server in scripting.

If you are running on a z/OS platform, this name is sometimes referred to as the long name.

Data type	String
Default	server1

Short name:

Specifies the short name of the server and must be unique within a cell. This field only applies to the z/OS platform. The short name is also the default z/OS job name and identifies the server to the native facilities of the operating system, such as Workload Manager (WLM), Automatic Restart Manager, SAF (for example, RACF), started task control, and others.

If you specify a short name, the name:

- Must be one to eight characters in length. By default, WebSphere Application Server for z/OS assumes you are using a 7-character server short name (JOBNAME). If your naming standards require 8 characters, you can lengthen the 7-character server short name to 8 characters.
- Must contain only alpha-numeric or national language characters.
- Cannot start with a number.
- Must be unique in the cell

If you do not specify a short name, the system assigns a default short name that is automatically unique within the cell. You can change the generated short name to conform with your naming conventions.

Run in development mode:

Enabling this option may reduce the startup time of an application server. This may include JVM settings such as disabling bytecode verification and reducing JIT compilation costs. Do not enable this setting on production servers. This setting is only available on application servers running WebSphere Application Server Version 6.0 and later.

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 will not be started in development mode. Setting this option to `true` specifies that the server will be started in development mode (with settings that will speed server startup time).

Data type	Boolean
Default	false

Parallel start:

Select this field to start the server on multiple threads. This might shorten the startup time.

Specifies that you want the server components, services, and applications to start in parallel rather than sequentially.

The default setting for this option is `true`, which indicates that the server be started using multiple threads. Setting this option to `false` specifies that the server will not be started in using multiple threads (which may lengthen startup time).

Note that the order in which the applications start depends on the weights you assigned to each them. Applications that have the same weight are started in parallel. You set an application's weight with the *Starting weight* option on the **Applications > Enterprise Applications >** *application_name* page of the Administrative Console.

Data type	Boolean
Default	true

Class loader policy:

Select whether there is a single class loader to load all applications or a different class loader for each application.

Class loading mode:

Specifies whether the class loader should search in the parent class loader or in the application class loader first to load a class. The standard for Developer Kit class loaders and WebSphere Application Server class loaders is `Parent first`.

This field only applies if you set the Class loader policy field to `single`.

If you select `Parent last`, your application can override classes contained in the parent class loader, but this action can potentially result in ClassCastException or linkage errors if you have mixed use of overridden classes and non-overridden classes.

Access to internal server classes:

Specifies whether the applications can access many of the server implementation classes.

If you select `Allow`, applications can access many of the server implementation classes. If you select `Restrict`, applications can not access many of the server implementation classes. The applications get a ClassNotFoundException if they attempt to access those classes.

Applications typically use the supported APIs and do not need to access system internals.

Process Id:

The native operating system's process ID for this server.

The process ID property is read only. The system automatically generates the value.

Cell name:

The name of the cell in which this server is running.

The Cell name property is read only.

Node name:

The name of the node in which this server is running.

The Node name property is read only.

State:

The run-time execution state for this server.

The State property is read only.

Ports collection:

Use this page to view and manage communication ports used by run-time components running within a process. Communication ports provide host and port specifications for a server.

To view this administrative console page, click **Servers** > **Application Servers** > *server_name* > **Communications** > **Ports**.

This page displays only when you are working with ports for application servers.

Port Name:

Specifies the name of a port. Each name must be unique within the server.

Host:

Specifies the IP address, domain name server (DNS) host name with domain name suffix, or just the DNS host name, used by a client to request a resource (such as the naming service, or administrative service).

Port:

Specifies the port for which the service is configured to accept client requests. The port value is used in conjunction with the host name.

Transport Details:

Provides a link to the transport chains associated with this port. If no transport chains are associated with this port, the string ″No associated transports″ appears in this column.

Ports settings:

Use this to view and change the configuration for a communication port used by run-time components running within a process. A communication port provides host and port specifications for a server.

For the z/OS platform, you can view this administrative console page by clicking one of the following paths:

- **Servers** > **Application Servers** > *server_name* > **Ports** > *end_point_name*
- **Servers** > **JMS Servers** > *server_name* > **Ports** > *end_point_name*

Port Name:

Specifies the name of the port. The name must be unique within the server.

Note that this field displays only when you are defining a port for an application server. You can select either:

Well-known Port
When you select this option , you can select a previously defined port from the drop down list

User-defined Port
When you select this option, you must create a port with a new name by entering the name of the new port in the text box

Data type	String

The following ports apply to the z/OS platform only.

End point	Description
JMSSERVER Queued Address	Specifies the host and port number used to configure the WebSphere JMS Provider topic connection factory. The JMS Server Queued Address port is the listener port used for full-function JMS-compliant, publish/subscribe support. The default Queued Address port number is 5558. Since the queue manager and queue broker for the WebSphere JMS Provider are configured outside the administrative console, changes to this port require corresponding configuration changes to the queue manager and queue broker.
JMSSERVER Direct Address	Specifies the host and port number used to configure the WebSphere JMS Provider topic connection factory. The JMS Server Direct Address port is the listener port used for direct TCP/IP connection (non-transactional, nonpersistent, and nondurable subscriptions only) for publish/subscribe support. The default Direct Address port number is 5559. Since the queue manager and queue broker for the WebSphere JMS Provider are configured outside the administrative console, changes to this port require corresponding configuration changes to the queue manager and queue broker.

Host:

Specifies the IP address, domain name server (DNS) host name with domain name suffix, or just the DNS host name, used by a client to request a resource (such as the naming service, administrative service, or JMS broker).

For example, if the host name is `myhost`, the fully qualified DNS name can be `myhost.myco.com` and the IP address can be `155.123.88.201`.

Host names on the ports can be resolvable names or IP addresses. The server will bind to the specific host name or IP address that is supplied. That port will only be accessible through the IP address that is resolved from the given host name or IP address. The IP address may be of the IPv4 (Internet Protocol Version 4) format for all platforms, and IPv6 (Internet Protocol Version 6) format on specific operating systems where the server supports IPv6.

Data type	String
Default	* (asterisk)

Port:

Specifies the port for which the service is configured to accept client requests. The port value is used in conjunction with the host name.

Port numbers in the server can be reused among multiple ports as long as they have host names that resolve to unique IP addresses and there is not a port with the same port number and a wildcard (*) host name. A port number is valid in the range of 0 and 65535. 0 specifies that the server should bind to any ephemeral port available.

Important: Port sharing cannot be created using the administrative console. If you need to share a port, you must use wsadmin commands to define that port. You must also make sure that the same discrimination weights are defined for all of the transport channels associated with that port.

Protocol channels only accept their own protocol. However, application channels usually accept anything that reaches them. Therefore, for application channels, such as WebContainer or Proxy, you should specify larger discrimination weights when sharing levels with protocol channels, such as HTTP or SSL. The one exception to this rule is if you have application channels that perform discrimination tests faster than the protocol channels. For example, a JFAP channel is faster at deciding on a request than the SSL protocol channel, and should go first for performance reasons. However, the WebContainer and Proxy channels must always be last because they accept everything that is handed to them.

Data type	Integer
Default	None

The following table lists server endpoints and their respective port ranges. In contrast to a distributed platform environment, on the z/OS platform, the ORB_LISTENER_ADDRESS and the BOOTSTRAP_ADDRESS must specify the same port.

Endpoint (port)	Acceptable values for the port field
BOOTSTRAP_ADDRESS	1 - 65536
CSIV2_SSL_MUTUALAUTH_LISTENER_ADDRESS	Not supported on the z/OS platform
CSIV2_SSL_SERVERAUTH_LISTENER_ADDRESS	Not supported on the z/OS platform
DCS_UNICAST_ADDRESS	1 - 65536
DRS_CLIENT_ADDRESS	1 - 65536
ORB_LISTENER_ADDRESS	1 - 65535
SAS_SSL_SERVERAUTH_LISTENER_ADDRESS	Not supported on the z/OS platform
SIB_ENDPOINT_ADDRESS	1 - 65536
SIB_ENDPOINT_SECURE_ADDRESS	1 - 65536
SIB_MQ_ENDPOINT_ADDRESS	1 - 65536
SIB_MQ_ENDPOINT_SECURE_ ADDRESS	1 - 65536
SOAP_CONNECTOR_ ADDRESS	1 - 65536
WC_adminhost	1 - 65536
WC_adminhost_secure	1 - 65536
WC_defaulthost	1 - 65536
WC_defaulthost_secure	1 - 65536
ORB_SSL_LISTENER_ADDRESS	0 - 65535 (0 specifies that the server should bind to any ephemeral port that is available.)

Custom property collection:

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.

Name:

Specifies the name (or key) for the property.

Each property name must be unique. If the same name is used for multiple properties, the value specified for the first property that has that name is used.

Do not start your property names with `was.` because this prefix is reserved for properties that are predefined in the application server.

Value:

Specifies the value paired with the specified name.

Description:

Provides information about the name-value pair.

Custom property settings:

Use this page to configure 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. Defining a new property enables you to configure a setting beyond that which is available in the administrative console.

To view this administrative console page, click one of the following paths:

- For an application server, click **Servers > Application Servers >** *server_name*. Then, under Server Infrastructure, click **Administration** > **Custom Properties** .
- For a JMS server, click **Servers** > **JMS Servers** > *server_name*. Then, under Server Infrastructure, click **Administration** > **Custom Properties** .
- For a deployment manager, click For a deployment manager, click **System administration > Deployment manager**, and then under Server Infrastructure, click > **Java and Process Management > Process Definition > Java Virtual Machine > Custom Properties**.

You can then click **New** to, create a new custom property, click on the name of an existing property to change its settings, or click **Delete** to delete an existing property.

Name:

Specifies the name (or key) for the property.

Each property name must be unique. If the same name is used for multiple properties, the value specified for the first property that has that name is used.

Do not start your property names with `was.` because this prefix is reserved for properties that are predefined in WebSphere Application Server.

Data type	String

Value:

Specifies the value paired with the specified name.

Data type	String

Description:

Provides information about the name and value pair.

Data type	String

Native processes:

Use this page to view and modify properties of the JMS Integral Provider native processes.

To view this administrative console page, click **Servers** > **Application Servers** > *server name*. Then, under Server Infrastructure, click **Administration** > **Server Components** > **JMS Servers**.

Short name:

Specifies the short name of the JMS queue manager.

The name is 1-4 characters, alpha-numeric or national language. It cannot start with a numeric.

The system assigns a unique default short name for the JMS queue manager.

Data type	String

Command Prefix:

Specifies the subsystem command prefix for the JMS queue manager.

This field is read only because it is only configured through the WebSphere ISPF Customization Dialog.

Data type	String

Server component collection:

Use this page to view information about and manage server component types such as application servers, messaging servers, or name servers.

To view this administrative console page, click **Servers > Application Servers >** *server_name*. Then, under Server Infrastructure, click **Administration > Server Components**.

Type:

Specifies the type of internal server.

Server component settings:

Use this page to view or configure a server component instance.

To view this administrative console, click **Servers > Application Servers >** *server_name*. Then, under Server Infrastructure, click **Administration > Server Components >** *server_component_name*.

Name:

Specifies the name of the component.

Data type	String

Initial State:

Specifies the desired state of the component when the server process starts. The options are: *Started* and *Stopped*. The default is *Started*.

Data type	String
Default	Started

Server instance settings:

Use this page to view and manage z/OS servant instance settings. These settings control the number of servant processes that are allowed.

To view this page, in a z/OS administrative console, click **Servers** > **Application Servers** > *server name*. Next, under Server Infrastructure, click **Java and Process Management > Server Instance**.

Multiple Instances Enabled:

Specifies whether multiple servant process instances are enabled for this server.

When the Multiple Instances Enabled setting is disabled, the server has exactly one servant process instance. When the Multiple Instances Enabled setting is enabled, the server has the following:
- A minimum number of servant process instances as specified on the Minimum Number of Instances setting
- A maximum number of servant process instances as specified on the Maximum Number of Instances setting

The z/OS Workload Manager dynamically determines the actual number of servant process instances.

Minimum Number of Instances:

Specifies the minimum number of servant process instances to activate.

Data type	Integer
Range	1 to 20, inclusive

Maximum Number of Instances:

Specifies the maximum number of servant process instances to activate.

Data type	Integer
Range	Any value. If zero is specified, the number of instances is unlimited.

Workload profile:

Specifies the server workload profile, which can be ISOLATE, IOBOUND, CPUBOUND, or LONGWAIT

The workload profile controls workload-pertinent decisions that are made by the WebSphere Application Server for z/OS run time, such as the number of threads used in the servant. The default value is IOBOUND, which is the appropriate value for most applications. Use one of the other values when your application requires more threads.

Workload profile	Number of Threads	Description

ISOLATE	1	Specifies that the servants are restricted to a single application thread. Use ISOLATE to ensure that concurrently dispatched applications do not run in the same servant. Two requests processed in the same servant can cause one request to corrupt another.
IOBOUND	MIN(30, MAX(5,(Number of CPUs*3)))	Specifies more threads in applications that perform I/O-intensive processing on the z/OS operating system. The calculation of the thread number is based on the number of CPUs. IOBOUND is used by most applications that have a balance of CPU intensive and remote operation calls. A gateway or protocol converter are two examples of applications that use the IOBOUND profile.
CPUBOUND	MAX((Number of CPUs-1),3)	Specifies that the application performs processor-intensive operations on the z/OS operating system, and therefore would not benefit from more threads than the number of CPUs. The calculation of the thread number is based on the number of CPUs. Use the CPUBOUND profile setting in CPU intensive applications, like XML parsing and XML document construction, where the vast majority of the application response time is spent using the CPU.
LONGWAIT	40	Specifies more threads than IOBOUND for application processing. LONGWAIT spends most of its time waiting for network or remote operations to complete. Use this setting when the application makes frequent calls to another application system, like Customer Information Control System (CICS) screen scraper applications, but does not do much of its own processing.

Note: Number of CPUs is the number of CPUs online when the controller comes up.

You can look at message BBOO0234I in the servant job log to check the number of worker threads.

Generic server settings

Use this page to view or change the settings of a generic server.

A generic server is a server that is managed in the WebSphere Application Server administrative domain, although it is not a server that is supplied by the WebSphere Application Server product. The generic server can be any server or process that is necessary to support the Application Server environment, including a Java server, a C or C++ server or process, or a Remote Method Invocation (RMI) server.

To view this administrative console page, click **Servers** > **Generic Servers** > *server_name*.

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 the server is running.

Name:

Specifies a logical name for the generic server.

Generic server names must be unique within a node. For multiple nodes within a cluster, you can have different generic servers with the same server name as long as the server and node pair are unique. For example, a server named server1 in a node named node1 in the same cluster with a server named server1 in a node named node2 is allowed. Configuring two servers named server1 in the same node is not allowed. WebSphere Application Server uses the server name for administrative actions, such as referencing the server in scripting.

It is highly recommended that you use a naming scheme that makes it easy to distinguish your generic application servers from regular WebSphere Application Servers. This will enable you to quickly determine whether to use the Terminate or Stop button in the administrative console to stop a specific application server.

You must use the Terminate button to stop a generic application server.

Data type	String
Default	

Changing the values of variables referenced in BBOM0001I messages

BBOM0001I messages are issued during server startup, and indicate the WebSphere Application Server configuration settings. Unless otherwise indicated, these variables apply to all servers (deployment managers, node agents, JMS servers, and application servers).

Some of these setting can not be changed. The ones you can change, must be changed using the administrative console. Changes made directly to the server's was.env file will be overridden the next time the server is started.

Use the following table to determine which WebSphere Application Server variable, custom property or administrative console field must be updated to change the value of a specific internal variable.

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields

Internal variable name	**How to change indicated value**	Comments
cell_name	User cannot change.	Initially specified during installation and customization.
cell_short_name	User cannot change.	Initially specified during installation and customization.
client_protocol _resolve_name	User cannot change.	No longer used. Message will not appear for V5.1 and higher.
client_protocol _resolve_port	User cannot change.	No longer used. Message will not appear for V5.1 and higher.

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued)

client_ras_ logstreamname	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **client_ras_logstreamname** property and specify a different value.	See Application server z/OS custom properties for additional information.
com_ibm_security_ SAF_EJBROLE_Audit _Messages_Suppress	In the administrative console,click **Security** > **Global Security**. Under Additional Properties. click **Custom Properties** > **New**. Add the **com_ibm_security_SAF_EJBROLE_Audit_Messages_Suppress** property and specify a different value.	
com_ibm_DAEMON_ claim_ssl_sys_v3_ timeout	In the administrative console, click **System Administration > Node groups > sysplex node group > z/OS Location Service**.	Select an existing alias or create a new SSL Configuration Repertoire.
com_ibm_DAEMON_ claimClient Authentication	In the administrative console, click **System Administration > Node groups > sysplex node group > z/OS Location Service**.	Select an existing alias or create a new SSL Configuration Repertoire.
com_ibm_DAEMON_ claimKeyringName	In the administrative console, click **System Administration > Node groups > sysplex node group > z/OS Location Service**.	Select an existing alias or create a new SSL Configuration Repertoire.
com_ibm_DAEMON_ claimSecurityCipher SuiteList	In the administrative console, click **System Administration > Node groups > sysplex node group > z/OS Location Service**.	Select an existing alias or create a new SSL Configuration Repertoire.
com_ibm_DAEMON_ claimSecurityLevel	In the administrative console, click **System Administration > Node groups > sysplex node group > z/OS Location Service**.	Select an existing alias or create a new SSL Configuration Repertoire.
com_ibm_HTTP_claim _ssl_sys_v2_timeout	User cannot change.	This variable has been deprecated.
com_ibm_HTTP_claim _ssl_sys_v3_timeout	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings > Web container** > **HTTP transports** > *ssl_transport*.	Transport option only appears if you have an transport defined for your system.
com_ibm_HTTP_claim _sslEnabled	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings > Web container** > **HTTP transports** > *ssl_transport*.	Transport option only appears if you have an transport defined for your system.
com_ibm_HTTP_claim ClientAuthentication	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings > Web container** > **HTTP transports**.	Transport option only appears if you have an transport defined for your system.
com_ibm_HTTP_claim KeyringName	In the administrative console, click **Server** > **Application Server** > *server* > **Web Container Settings > Web container** > **HTTP transports**.	Transport option only appears if you have an transport defined for your system.
com_ibm_HTTP_claim SecurityCipherSuite List	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings > Web container** > **HTTP transports**.	Transport option only appears if you have an transport defined for your system.
com_ibm_HTTP_claim SecurityLevel	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings > Web container** > **HTTP transports**.	Transport option only appears if you have an transport defined for your system.

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued)

com_ibm_Server_ Security_Enabled	In the administrative console, click **Servers** > **Application Servers** > *server* > **Security** > **Sever Security**.	Select an existing alias or create a new SSL Configuration Repertoire. This setting overrides the setting specified using the **Security** > **Global Security** > **enabled/disabled** field in the administrative console.
control_region_ classpath	User cannot change. However an additional path can be appended to the specified path. To do this, in the administrative console click **Servers** > **Application Servers** > *server* > **Java and Process Management > Process Definitions** > **Java Virtual Machine** and specify the classpath to be appended.	Specifies the classpath used by the conroller's JVM.
control_region_jvm _localrefs		Should only be used under the direction of IBM support.
control_region_jvm _logfile	User cannot change.	Specifies the file to which the conroller's JVM will write messages.
control_region_jvm _properties_file	User cannot change.	Is dynamically created.
control_region_ libpath	In the administrative console, click **Server** > **Application Server** > *server* > **Java and Process Management > Process Definitions**, and specify the libpath.	
control_region_mdb _request_timeout (application server only)	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **control_region_mdb_request_timeout** property and specify a different value.	See Application server z/OS custom properties for additional information.
control_region_security _enable_trusted_ applications (application server only)	In the administrative console, click **Security** > **Global Security** > **Custom Properties**. Select the **EnableTrustedApplications** property and specify a new value.	
control_region_ssl_ thread_pool_size		Should only be changed under the direction of IBM Support personnel.
control_region_thread _pool_size		Should only be changed under the direction of IBM Support personnel.
control_region_thread _stack_size		Should only be changed under the direction of IBM Support personnel.
control_region_use _java_g	In the administrative console, click **Servers** > **Application Servers** > *server* > **Java and Process Management > Process Definitions Java Virtual Machine** and select **Debug Mode**.	Indicates if conroller's JVM should use the debug JVM (java_g). Should only be changed under the direction of IBM Support personnel.

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued)

control_region_wlm_dispatch_timeout (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server*. Under Container Settings, click **Container Services > ORB Service**. Specify a different value in the **WLM Timeout** field.	
daemon_group_name	User cannot change.	
daemon_protocol_iiop_listenIPAddress	In the administrative console, click **Environment** > **WebSphere Variables** > **New**. Add the **DAEMON_protocol_iiop_listenIPAddress**. In the Value field, type ***** to specify bind all, or the IP name to specify bind-specific support.	Allows you to restrict the IP addresses to which the location service daemon binds. You set this variable in your cell-level variables.xml file.
daemon_start_command	User cannot change.	
daemon_start_command_args	User cannot change.	
daemon_wlmable	User cannot change.	
daemonInstanceName	User cannot change.	
daemonName	User cannot change.	
nls_language	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **nls_language** property and specify a different value.	See Application server z/OS custom properties for additional information.
node_name	User cannot change.	
node_short_name	User cannot change.	
nonauthenticated_clients_allowed	In the administrative console, click **Security > Global security** . Under Authentication, click **Authentication Protocol** > **zSAS authentication**.	
protocol_accept_http_work_after_min_srs	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_accept_http_work_after_min_srs** property and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_bboc_log_response_failure	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_bboc_log_response_failure** property and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_bboc_log_return_exception	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_bboc_log_return_exception** property and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_giop_level_highest	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_giop_level_highest** property and specify a different value.	See Application server z/OS custom properties for additional information.

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued)

protocol_http_backlog	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_http_backlog** property and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_http_large_ data_inbound_buffer	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_http_large_data_inbound_buffer** property and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_http_large_ data_response_buffer	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_http_large_data_response_buffer** property and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_http_max _connect_backlog	User cannot update.	No longer used.
protocol_http_max_ keep_alive_ connections	User cannot update.	No longer used
protocol_http_timeout _output (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings > Web container > HTTP transport >** *non_ssl_transport* > **Custom Properties**. Select the **ConnectionResponseTimeout** property and specify a new value.	See HTTP transport custom properties for additional information.
protocol_http_timeout _output_recovery (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_http_timeout_output_recovery** property and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_http_ transactionClass	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_http_transactionClass** property and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_http_transport _class_mapping_file	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings > Web container** > **z/OS additional Settings**.	**Important:** This variable is deprecated. When possible, use a workload classification document file instead of this variable. See "Classifying z/OS workload" on page 381 for more information.
protocol_https_backlog	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_https_backlog** property and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_https_max_ connect_backlog	User cannot change.	No longer used.
protocol_https_max_ keep_alive_ connections	User cannot change.	No longer used.

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued)

protocol_https_timeout _output (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings** > **Web container > HTTP transports** > *ssl_transport* > **Custom Properties**. Select the **ConnectionResponseTimeout** property and specify a new value.	See HTTP transport custom properties for additional information.
protocol_https_timeout _output_recovery (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_https_timeout_ output_recovery** property and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_https_ transactionClass	User cannot change. The value specified for the protocol_http_transport_class_mapping_file variable is also used for this variable.	
protocol_iiop_backlog _ssl	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_iiop_backlog_ssl** property and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_iiop_backlog	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_iiop_backlog** property and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_iiop_daemon _listenIPAddress	In the administrative console, click **System Administration > Node groups > sysplex node group > z/OS Location Service**. Specify the new IP address.	
protocol_iiop_daemon _port	In the administrative console, click **System Administration > Node groups > sysplex node group > z/OS Location Service**.	
protocol_iiop_daemon _port_ssl	In the administrative console, click **System Administration > Node groups > sysplex node group > z/OS Location Service**.	
protocol_iiop_no_ local_copies	In the administrative console, click **Servers** > **Application Servers** > *server*. Under Container Settings, click **Container Services > ORB Service**.	
protocol_iiop_propagate _unknown_service_ctxs	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_iiop_propagate_unknown_service_ctxs** property and specify a different value.	See Application server z/OS custom properties for additional information.
ras_debugEnabled		Should only be changed under the direction of IBM Support personnel.
ras_default_msg_dd	In the administrative console, click **Environment** > **Manage WebSphere Variables** > **New**. Add the **ras_default_msg_dd** variable.	
ras_dumpoptions_ dumptype		Should only be changed under the direction of IBM Support personnel.
ras_hardcopy_ msg_dd	In the administrative console, click **Environment** > **WebSphere Variables** > **New**. Add the **ras_hardcopy_msg_dd** variable.	
ras_log_ logstreamName	In the administrative console, click **Environment** > **WebSphere Variables** > **New**. Add the **ras_log_logstreamName** variable.	

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued)

ras_minorcode_action		Should only be changed under the direction of IBM Support personnel.
ras_time_local	In the administrative console, click **Environment** > **WebSphere Variables** > **New**. Add the **ras_time_local** variable.	
ras_trace_basic		Should only be changed under the direction of IBM Support personnel.
ras_trace_ctraceParms		Should only be changed under the direction of IBM Support personnel.
ras_trace_ defaultTracingLevel		Should only be changed under the direction of IBM Support personnel.
ras_trace_detail		Should only be changed under the direction of IBM Support personnel.
ras_trace_exclude _specific		Should only be changed under the direction of IBM Support personnel.
ras_trace_minor CodeTraceBacks		Should only be changed under the direction of IBM Support personnel.
ras_trace_ outputLocation	In the administrative console, click **Environment** > **WebSphere Variables** > **New**. Add the **ras_trace_outputLocation** variable.	
ras_trace_specific		Should only be changed under the direction of IBM Support personnel.
ras_trace_BufferCount	In the administrative console, click **Environment** > **WebSphere Variables** > **New**. Add the **ras_trace_BufferCount** variable.	
ras_trace_BufferSize	In the administrative console, click **Environment** > **Manage ebSphere Variables** > **New**. Add the **ras_trace_BufferSize** variable.	
read_license_ agreement	User cannot change.	Indicates that the server's initialization code will verify license agreement. The default is 1. If this variable is not set to 1, the server will not initialize.
security_SMF_record _first_auth_user	In the administrative console, click **Server** > **Application ServersSet** > **server name** > **Custom Properties**. Set **security_SMF_record_first_auth_user** equal to 1 to fill the SM120CRE field with the ID of the first authenticated user. By default, the property is set to 0, and the SM120CRE field is filled with the ID under which the server activity began. This is often the guest ID.	If no authentication for a request is required, then the SM120CRE field will NOT contain an authenticated user ID for that request. Instead, it will contain an unauthenticated ID, typically the guest account ID or the WebSphere server ID.
security_sslKeyring	User cannot change.	No longer used.

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued)

security_zOS_ domainName	User cannot change.	Set during customization.
security_zOS_ domainType	User cannot change.	Set during customization.
security_zSAS_ ssl_repertoire	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server Security** > **zSAS Transport** > **SSL Setttings**. Select a different repertoire.	
security_ EnableRunAsIdentity	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **security_EnableRunAsIdentity** property and specify a different value.	See Application server z/OS custom properties for additional information.
security_sslType1	User cannot change.	No longer used.
server_configured _system_name	User cannot change.	Specifies the name of the system to which the server instance was originally configured.
server_generic_short _name	If the server is clustered, this value is the cluster's short name. If the server is not clustered, this value is the value specified on the server custom property, **ClusterTransitionName**. Either way, this value can be changed using the administrative console.	
server_generic_uuid	User cannot change.	Specifies the unique identifier for this server
server_region_ classpath (application server and deployment manager only)	User cannot change. However an additional path can be appended to the specified path. To do this, in the administrative console click **Servers** > **Application Servers** > *server*. Under Server Infrastructure click **Java and Process Management > Process Definitions** > **Servant** > **Java Virtual Machine** > **Classpath**, and specify the classpath to be appended.	Specifies the classpath used by the servant region's JVM.
server_region_ dynapplenv_jclparms (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server*. Under Server Infrastructure click **Java and Process Management > Process Definitions** > **servant** > **Start Command Args**. Specify the new parameters.	When dynamic applenv is being used (instead of the same content existing in static definition of WLM panels), this variable specifies the JCL parameters provided to the servant region when WLM starts this servant region.
server_region_ dynapplenv_jclproc (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server*. Under Server Infrastructure click **Java and Process Management > Process Definitions** > **servant** > **Start Command**. Specify the new JCL procedure.	When dynamic applenv is being used., this variable specifies the name of the JCL procedure for a servant region when WLM starts this servant region.
server_region_jvm _localrefs (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **server_region_jvm_localrefs** property and specify a different value.	See Application server z/OS custom properties for additional information.

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued)

server_region_jvm _logfile (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server* > **Custom Properties** > **New**. Add the **server_region_jvm_logfile** property and specify a different HFS file.	Specifies the HFS file that JNI debug messages are written to.
server_region_jvm _properties_file (application server and deployment manager only)	User cannot change	File is dynamically created.
server_region_libpath (application server and deployment manager only)	**Servers** > **Application Servers** > *server*. Under Server Infrastructure click **Java and Process Management > Process Definitions** > **Servant** > **Environment Entries** > **LIBPATH**.	Specifies the libpath for the servant region's JVM
server_region_recycle _count (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **server_region_recycle_count** property and specify a different value.	See Application server z/OS custom properties for additional information.
server_region_thread _stack_size (application server and deployment manager only)		Should only be changed under the direction of IBM Support personnel.
control_region_ timeout_delay (application server and deployment manager only)	In the administrative console, click **Servers > Application Servers** > *server* **> Custom Properties > New**. Add the **control_region_timeout_delay** property and specify a different value.	See Configuring server properties for additional information.
server_region_use _java_g (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server*. Under Server Infrastructure click **Java and Process Management > Process Definitions** > **Servant** > **Java Virtual Machine** and select **Debug Mode**.	Indicates if the servant region's JVM should use the debug JVM (java_g). Should only be used under the direction of IBM support.
server_region_ workload_profile (application server and deployment manager only)	n the administrative console, click **Servers** > **Application Servers** > *server*. Under Container Settings, click **Container Services > ORB Service**.	
server_specific_name	User cannot change.	
server_specific_short _name	In the administrative console, click **Servers** > **Application Servers** > *server* > **Short Name**.	
server_specific_uuid	User cannot change.	Specifies the unique identifier for this server

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued)

server_start_wait_for _initialization_Timeout	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **server_start_wait_for_initialization_Timeout** property and specify a different value. For a node agent, in the administrative console, click **System Administration > Node Agents >** *nodeagent* > **Administration Services > Custom Properties**. Add the **server_start_wait_for_initialization_Timeout** property and specify a different value.	See Application server z/OS custom properties for additional information.
server_SMF_container _activity_enabled (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server* > **Custom Properties** > **New**. Add the **server_SMF_container_activity_enabled** property and specify a different value.	See the section ″Using the WebSphere Application Server administrative console to enable properties for specific SMF record types″ in the *Troubleshooting and support* PDF for more information.
server_SMF_container _interval_enabled (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server*. Under Server Infrastructure, click **Administration > Custom Properties** > **New**. Add the **server_SMF_container_interval_enabled** property and specify a different value.	See the section ″Using the WebSphere Application Server administrative console to enable properties for specific SMF record types″ in the *Troubleshooting and support* PDF for more information.
server_SMF_interval _length (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server*. Under Server Infrastructure, click **Administration > Custom Properties** > **New**. Add the **server_SMF_interval_length** property and specify a different value.	See the section ″Using the WebSphere Application Server administrative console to enable properties for specific SMF record types″ in the *Troubleshooting and support* PDF for more information.
server_SMF_server _activity_enabled (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server* . Under Server Infrastructure, click **Administration > Custom Properties** > **New**. Add the **server_SMF_server_activity_enabled** property and specify a different value.	See the section ″Using the WebSphere Application Server administrative console to enable properties for specific SMF record types″ in the *Troubleshooting and support* PDF for more information.

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued)

server_SMF_server _interval_enabled (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server*. Under Server Infrastructure, click **Administration > Custom Properties** > **New**. Add the **server_SMF_server_interval_enabled** property and specify a different value.	See the section ″Using the WebSphere Application Server administrative console to enable properties for specific SMF record types″ in the *Troubleshooting and support* PDF for more information.
server_SMF_web_ container_activity_ enabled	User cannot change.	No longer used.
server_SMF_web_ container_interval_ enabled	User cannot change.	No longer used.
serverRegionid	User cannot change.	No longer used.
transaction_ defaultTimeout (application server only)	In the administrative console, click **Servers** > **Application Servers** > *server* > **Container Services > Transaction Service** > **Transaction lifetime timeout**.	The maximum duration, in seconds, for transactions on this application server. Any transaction that is not requested to complete before this timeout will be rolled back. Default is 120.
transaction_ maximumTimeout (application server only)	In the administrative console, click **Servers** > **Application Servers** > *server* > **Container Services > Transaction Service** > **Maximum Transaction Timeout** and specify a different value.	The maximum duration, in seconds, that transactions propagated into the server or transactions started by BMT components from within the server will be allowed to execute. Any transaction that is not requested to complete before this timeout will be rolled back. Default is 300.
transaction_ recoveryTimeout (application server only)	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **transaction_recoveryTimeout** property and specify a different value.	See Application server z/OS custom properties for additional information.
was_env_file	User cannot change.	File is dynamically created.
wlm_dynapplenv_single _server (application server and deployment manager only)	For an application server, in the administrative console, click **Servers** > **Application Servers** > *server*. Under Server Infrastructure, click **Java and Process Management > Server Instance**. For a deployment manager, in the administrative console, click **System administration** > **deployment manager**. Under Server Infrastructure, click **Java and Process Management > Server Instance**.	

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued)

wlm_ maximumSRCount (application server and deployment manager only)	For an application server, in the administrative console, click **Servers** > **Application Servers** > *server*. Under Server Infrastructure, click **Java and Process Management > Server Instance**. For a deployment manager, in the administrative console, click **System administration** > **deployment manager**. Under Server Infrastructure, click **Java and Process Management > Server Instance**.	
wlm_ minimumSRCount (application server and deployment manager only)	For an application server, in the administrative console, click **Servers** > **Application Servers** > *server*. Under Server Infrastructure, click **Java and Process Management > Server Instance**. For a deployment manager, in the administrative console, click **System administration** > **deployment manager**. Under Server Infrastructure, click **Java and Process Management > Server Instance**.	

After you have determined which WebSphere Application Server variable, custom property or administrative console field must be updated to change the value of a specific internal variable:

1. Navigate to the appropriate administrative console panel. The ″How to change indicated value″ column describes how to navigate within the administrative console to the appropriate panel for changing a specific internal variable setting. For example, to change the setting of the wlm_maximumSRCount variable for an application server, click **Servers** > **Application Servers** > *server*, and then under Server Infrastructure, click **Java and Process Management > Server Instance**.
2. Update the variable, custom property, or administrative console field with the new value.
3. Save your changes.

Environment entries collection

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

To view this page, in the administrative console click **Proxy Servers >** *server_name*, and then under Server Infrastructure, click **Java and Process Management > Environment Entries**.

Name

Specifies the name (or key) for the environment entry. The name is a string that is used to set an internal system configuration environment entry.

Each environment entry name must be unique. If the same name is used for multiple environment entries, the value specified for the first environment entry that has that name is used.

Do not start your environment entry names with `was.` because this prefix is reserved for environment entries that are predefined for WebSphere Application Server.

Value

Specifies the value paired with the specified name.

Description

Provides information about the name-value pair.

Environment entries settings

Use this page to configure arbitrary name-value pairs of data, where the name is an environment entry key and the value is a string value that can be used to set internal system configuration environment entries. Defining a new environment entry enables you to configure a setting beyond that which is available in the administrative console.

To view this page, in the administrative console click **Proxy Servers >** *server_name*. Under Server Infrastructure, click **Java and Process Management > Environment Entries**. Then do one of the following:

- Click **New** to create a new environment entry.
- Click the name of an existing environment entry to change its settings,
- Select an existing environment entry and click **Delete** to delete that entry.

Name:

Specifies the name (or key) for the environment entry.

Each environment entry name must be unique. If the same name is used for multiple environment entries, the value specified for the first environment entry that has that name is used.

Do not start an environment entry name with `was.` because this prefix is reserved for environment entries that are predefined in WebSphere Application Server.

Data type	String

Value:

Specifies the value paired with the specified name.

Data type	String

Description:

Provides information about the name and value pair.

Data type	String

Starting an application server

Starting an application server starts a new server process based on the process definition settings of the current server configuration.

Before you start an application servers verify that all resource managers, such as DB2 and CICS, that your applications require are available. You must also start all prerequisite subsystems.

The node agent for the node on which the Application Server resides must be running before you can start the application server.

This procedure for starting a server also normally applies to restarting a server. The one exception might be if a server fails and you want the recovery functions to complete their processing prior to new work being started on that server. In this situation you must restart the server in recovery mode.

After you create a new application server definition, you can start, stop, or manage the new server using the administrative console, or you can use commands to perform these tasks for the new server.

After you start an Application Server, other processes might not immediately discover the running application server. Application servers are discovered by the node agent. However, node agents are discovered by the deployment manager. Even though node agents usually discover local application servers quickly, it might take a deployment manager up to 60 seconds to discover a node agent.

If you are using clusters, the **Initial State** property of the application server subcomponent (**Servers** > **Application servers** > *server_name* > **Administration** > **Server Components** > **Application Server**) is not intended to be used to control the state of individual servers in the cluster at the time the cluster is started. It is intended only as a way to control the state of the Application Server subcomponent of a server. It is best to start and stop the individual servers of a cluster using the Server options of the administrative console or command line commands (**startServer** and **stopServer**).

There are several options for starting an application server:

- You can use the administrative console:
  1. In the administrative console, click **Servers > Application servers** to determine the node agent on which the application server you are starting resides.
  2. Click **System administration > Nodes** and make sure the node agent is running.

     If the node agent is not running, issue the **startNode** command and then issue the **startServer** command. Once a node agent completely stops running and remains stopped, you cannot remotely start the node agent from the Node Agents page in the administrative console. You must issue the **startNode** command to start the node agent on the node where it runs.

     See the *Using the administrative clients* PDF for more information on how to issue these commands.
  3. Click **Servers > Application servers** again and select the application server you want to start.
  4. Click **Start**. You can view the **Status** value and any messages or logs to make sure the application server starts.
- You can issue a startserver command.

  If the node agent for the node on which the application server resides is not running, run the startnode command and then run the startserver command.
- You can issue a start command from the MVS console. A typical WebSphere Application Server for z/OS run time includes two nodes:
  – Deployment Manager node. This includes a location service daemon and a Deployment Manager with a controller and any number of servants
  – Application Server node. This includes a location service daemon, a node agent with a controller, and an application server with a controller and any number of servants
  1. 1. To start a server, issue the following command all in uppercase and on a single line. (It is shown here on two lines for printing purposes.)

     ```
     S controlregionprocname, JOBNAME= server_shortname,
         ENV= cell_shortname.Node_shortname.Server_shortname
     ```

     where:

     controlregionprocname
     Is the JCL procedure name in the proclib that is used to start the server.

     server_shortname
     Is the short name of the server (or the step name used to start the proc). This allows you to identify the address space that is running when you view it in the SDSF panels.

     cell_shortname.Node_shortname.Server_shortname
     The ENV variable is a concatenation of the cell short name, the node short name, and the server short name.

     For example:

     ```
     S BBO6ACR.BBOS001,ENV=SY1.SY1.BBOS001
     ```

The following messages indicate that the controller is up:

```
$HASP100 BBO6ACR  ON STCINRDR
$HASP373 BBO6ACR  STARTED
BBOO0001I WEBSPHERE FOR Z/OS CONTROL PROCESS BBODMNB/SY1/BBOC001/BBOS001
          IS STARTING.
IRR812I PROFILE BBO*.* (G) IN THE STARTED CLASS WAS USED TO START BBOS001S
        WITH JOBNAME BBOS001S.
$HASP100 BBOS001S ON STCINRDR
$HASP373 BBOS001S STARTED
+BBOO0004I WEBSPHERE FOR Z/OS SERVANT PROCESS BBODMNB/SY1/BBOC001/BBOS001
           IS STARTING.
+BBOO0020I INITIALIZATION COMPLETE FOR WEBSPHERE FOR Z/OS SERVANT PROCESS
           BBOS001.
BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR Z/OS CONTROL PROCESS
          BBOS001.
```

2. The controller automatically issues a command, similar to the following command, to start the daemon.

   ```
   S <dmn_proc>,JOBNAME=<dmn_jobname>,
          ENV=<cell_shortname.Node_shortname.daemon_instancename>
   ```

   Following is an example of the messages that are displayed during daemon startup:

   ```
   BBOO0001I WEBSPHERE FOR Z/OS CONTROL PROCESS BBODMNB/SY1/BBOC001/BBOS001
             IS STARTING.
   IRR812I PROFILE BBO*.* (G) IN THE STARTED CLASS WAS USED TO START BBO6DMN
           WITH JOBNAME BBO6DMN.
   $HASP100 BBO6DMN  ON STCINRDR
   $HASP373 BBO6DMN  STARTED
   BBOO0007I WEBSPHERE FOR Z/OS DAEMON BBODMNB/SY1/BBODMNB/SY1 IS STARTING.
   IEC130I STEPLIB  DD STATEMENT MISSING
   ITT102I CTRACE WRITER BBOWTR IS ALREADY ACTIVE.
   BBOO0215I PRODUCT 'WAS FOR Z/OS' SUCCESSFULLY REGISTERED WITH IFAED SERVICE.
   BBOO0015I INITIALIZATION COMPLETE FOR DAEMON SY1.
   ```

   WLM issues a command, similar to the following command, to start servant address spaces:

   ```
   S <Srv_Reg_Proc>,JOBNAME=<Server_shortname>,
              ENV=<Cell_shortname.Node_shortname.Server_shortname>
   ```

3. Determine if the daemon is up.

   See the *Using the administrative clients* PDF for more information. If the Daemon is up, use the administrative console to start the application server (see Step 1).

- **Optional:** Start an application server with tracing and debugging active.

  To start the Application Server with standard Java debugging enabled:

  1. In the administrative console, click **Servers > Application Servers**, click the application server whose processes you want to trace and debug, and then click **Java and Process Management > Process Definition > Java Virtual Machine**.
  2. On the Java virtual machine page, select the **Debug Mode** setting to enable the standard Java debugger. If needed, set debug arguments. Then, click **OK**.
  3. Save the changes to a configuration file.
  4. Stop the Application Server.
  5. Start the Application Server again as previously described.

After the server starts, install your applications.

You can use one of these same options to stop an application server.

Restarting an application server in recovery mode

When an application server instance with active transactions in progress restarts after a failure, the transaction service uses recovery logs to complete the recovery process. These logs, which each transactional resource maintains, are used to rerun any InDoubt transactions and return the overall system to a self-consistent state.

When you restart an application server in recovery mode:

- Transactional resources complete the actions in their recovery logs and then shut down. This action frees up any resource locks that the application server held prior to the failure.
- During the recovery period, only the subset of application server functions that are necessary for transactional recovery to proceed are available.
- The application server does not accept new work during the recovery process.
- The application server shuts down when the recovery is complete.

This recovery process begins as soon as all of the necessary subsystems within the application server are available. If the application server is not restarted in recovery mode, the application server can start accepting new work as soon as the server is ready, which might occur before the recovery work has completed.

Normally, this process is not a problem. However, situations exist when your operating procedures might not be compatible with supporting recovery work and new work simultaneously. For example, you might have a high availability environment where the work handled by the application server that failed is immediately moved to another application server. This backup application server then exclusively processes the work from the application server that failed until recovery has completed on the failed application server and the two application servers can be re-synchronized. In this situation, you might want the failing application server to only perform its transactional recovery process and then shut down. You might not want this application server to start accepting new work while the recovery process is taking place.

To prevent the assignment of new work to an application server that is going through its transaction recovery process, restart the application server in recovery mode.

When you restart a failed application server, the node agent for the node on which the failed application server resides must be running before you can restart that application server.

If you want to be able restart an application server in recovery mode, you must perform the following steps before a failure occurs, and then restart the application server to enable your configuration changes:

- If the server is monitored by a node agent, you must clear the Automatic restart option for that server. Clearing this option prevents the node agent from automatically restarting the server in normal mode, before you have a chance to start it in recovery mode.
  1. In the administrative console, click **Servers > Application Servers >** *server_name*.
  2. Under Server Infrastructure, click **Java and Process Management > Monitoring Policy**.
  3. Clear the Automatic restart option.
- If a catastrophic failure occurs that leaves InDoubt transactions, issue the **startServer** *server_name* **-recovery** command from the command line. This command restarts the server in recovery mode. You must issue the command from the *install_root*/profiles/*profile_name*/bin directory for the profile with which the server is associated.

The application server restarts in recovery mode, performs transactional recovery, and shuts down. Any resource locks that the application server held prior to the failure are released.

See the *Administering applications and their environment* PDF for information about the integrated high availability support for the transaction service subcomponent and how to configure it for peer recovery of transactions.

InFlight work and presumed abort mode:

Presumed abort mode is activated when a failure occurs before a distributed transaction starts to commit.

If you have a distributed transaction that spans several servers, transactional locks may be held by resource managers involved in that work. When a failure occurs before that distributed transaction has started to commit, WebSphere Application Server for z/OS and the resource managers go into presumed abort mode. In this mode, the resource managers abort (rollback) the transaction.

- The effect of a server failure or communications failure will vary depending on which server is running the work at the time of failure.
- An OTS timeout may be required to rollback the subordinate branches of the distributed transaction tree.

Example: A common case of this is when you have a server B Web client that is driving a session bean in the same server. That session bean has executed work against entity beans in servers C and D. All of the servers are involved in the same distributed, global transaction. Suddenly, server B fails while the session bean is InFlight (meaning it hadn't started to commit yet). Servers C and D are waiting for more work or the start of the two-phase commit protocol, but, while in this state, the transactional locks may still be held by the resource managers. So, the server roles are as follows:

- Server A: Servlet/JavaServer Page executed
- Server B: Session bean accessed
- Server C: Entity bean accessed
- Server D: Entity bean accessed

After the timeout occurs, because the session bean is InFlight at the time of the failure, WebSphere Application Server rolls back the transaction branch.

When local resource managers are involved, RRS ensures that they are called to perform presumed abort processing. When doing recovery, RRS works with the resource managers to ensure that the recovery is done properly. When a failure occurs while work is InFlight, RRS directs the resource managers involved in the local UR to rollback.

WebSphere Application Server for z/OS always assumes that there is recovery to do. Every time a server comes up, it does something different depending in which mode it is running:

- If the server is running in restart/recovery mode, WebSphere Application Server for z/OS checks to see whether there is any recovery required. If recovery is required, WebSphere Application Server for z/OS attempts to complete the recovery and either succeeds or terminates.
- If the server is running normally, the restart/recovery transaction does not have to complete before the server takes on new work. After the server determines what the restart work is, it begins to take in new work items. Processing of the restart/recovery transaction continues along with the processing of new work items.

IMS Connect considerations following server recovery:

After InDoubt and InFlight work completes, the WebSphere Application Server for z/OS server shuts down. A new application server configured for that system is then started up to accept new work. Special considerations must be taken if you are using IMS Connect after recovering to an alternate system.

After server recovery is completed, IMS Connect starts, but is not usable without some manual intervention. On the current IMS Connect WTOR perform the following commands `nn,viewhws` followed by `nn,viewhwsnn,opends` *XXX* where *XXX* is the IMS subsystem name displayed in the result of the `nn,viewhws` query. The IMS datastore needs to reflect 'active' status, as is shown in the following example:

```
*17 HWSC0000I *IMS CONNECT READY*  IMSCONN
 R 17,VIEWHWS
 IEE600I REPLY TO 17 IS;VIEWHWS
 HWSC0001I   HWS ID=IMSCONN     Racf=N
 HWSC0001I      Maxsoc=100  Timeout=12000
 HWSC0001I    Datastore=IMS      Status=ACTIVE
 HWSC0001I      Group=IMSGROUP Member=IMSCONN
 HWSC0001I      Target Member=IMSA
```

```
HWSC0001I    Port=9999     Status=ACTIVE
HWSC0001I      No active Clients
HWSC0001I    Port=LOCAL    Status=ACTIVE
HWSC0001I      No active Clients
```

After you complete the required manual intervention, IMS Connect is ready to handle new work on this server.

Detecting and handling problems with runtime components

You must monitor the status of runtime components to ensure that, once started, they remain operational as needed.

1. Regularly examine the status of runtime components.

   Browse messages displayed under WebSphere Runtime Messages in the status area at the bottom of the console. The runtime event messages, marked with a red X, provide detailed information on event processing.
2. If an application stops runningl, examine the status of the application If an application stops running when it should be operational, examine the status of the application on an Applications page and try restarting the application. If messages indicate that a server has stopped running, use the Application servers page to try restarting the server. If a cluster of servers stops running, use the Server Cluster page to try to restart the cluster. If the status of an application server is Unavailable, the node agent is not running in that node and you must restart the node agent before you can start the server.
3. If the runtime components do not restart, reexamine the messages and read information on problem determination to help you to restart the components.

Stopping an application server

Stopping an application server ends a server process based on the process definition settings in the current application server configuration.

There are several options for stopping an application server:

- You can stop the application server from the command line.

  You can use the stopserver or the stopmanager command.

  You should not use the `CANCEL appserver_proc_name` command to stop a server. Every time a server is started, a new temp directory is created off of the servant process token, such as *profile_root*/default/temp/*node_name*/*server_name*. When the server is cleanly stopped, these temp directories are normally removed. However, if the server is frequently not stopped cleanly, which happens if you cancel rather than stop the server, these temp directories are not removed and the HFS used for these temp directories eventually becomes full. You can also prevent this storage problem from occurring if you precompile your JavaServer pages when you install an application or if you use the JspBatchCompiler function to precompile them before they are invoked.
- You can use the administrative console to stop an application server:
  1. In the administrative console, click **Servers > Application Servers**.
  2. Select the application server that you want stopped and click **Stop**.
  3. Confirm that you want to stop the application server.
  4. View the **Status** value and any messages or logs to see whether the application server stops.

If you experience any problems shutting down a server, see the *Troubleshooting and support* PDF.

Automatically rejecting work requests when no servant is available to process these requests

When a controller determines that a servant has terminated, it normally cleans up any other work requests that were dispatched in that servant. If that servant was the last servant, new work requests are placed in the request queue until a servant is available. Depending on how long it takes for a servant to become available, these requests might terminate because the time allowed to process a request has expired. To prevent this from happening, you can change the configuration settings for an application server to prevent the controller from accepting new requests.

Controllers receive application requests on a continual basis and dispatch them to a servant for processing. When a system level problem, such as a database error, occurs, request processing stops and requests pile up in the queues between the controller and the servants. During the time it takes for a servant to become available, requests continue to pile up in the queues until they start to time out. When a timeout occurs, the request that timed out is removed from the queue.

When a new servant is ready to start accepting requests, the next request in the queue might be so close to timing out that the dispatch process for the request cannot complete and the servant again is terminated by timeout processing. Again requests accumulate in the queue until another new servant is ready and potentially the same timeout problem occurs. When this problem keeps reoccurring, it is sometimes referred to as a *bouncing servant* problem. You can handle this problem in one of the following ways:

- You can configure the server to automatically detect a no-server situation and stop taking requests until the minimum configured number of servants are ready to accept work. This is the simplest approach.
- You can create an automation routine to handle the problem if you are able to detect that you are having a system problem before servants are terminated because of timeouts. This automation routine can issue the `f` *server*, `pauselisteners` command to prevent requests from being accepted by this server. The routine must then detect when circumstances have changed and issue the `f` *server*, `resumelisteners` command when the detected system problem is resolved.
- You can configure the server to detect a no-server condition and stop taking requests, and create the previously discussed automation routine. The automation routine must recognize the different processing that can take place because the server is configured to detect a no-server condition:
  - If the last servant terminates even though the `f` *server*, `pauselisteners` command is issued, the server starts to reject all requests and issues message BBOO0299I. The server automatically starts to accept requests when the minimum number of servants, for which the server is configured, are ready to accept work. It also issues message BBOO0300I to indicate that requests are again being processed. Therefore, the automation routine must be sensitive to the fact that the server might have resumed accepting requests upon detecting that the minimal number of servants are available.
  - If the control_region_confirm_recovery_on_no_srs custom property is specified for the server, the server issues WTOR message BBOO0297A after it detects that the minimal number of servants, for which the server is configured, are ready to process new requests. You must enter a response to this message before the server actually starts to accept work.
  - If the automation routine prevents the server from terminating the servants because of timeout processing, it must also recognize when it is safe for the server to resume taking requests and issue the `f` *server*, `resumelisteners` command at that point in time. The automation routine can be set up to determine whether or not it needs to issue the `f` *server*, `resumelisteners` command based on whether or not the message BBOO0299I is issued. This message indicates that the server ran out of servants and is rejecting requests. This approach is the most complex, but provides the most flexibility.

To configure the server to handle no-server conditions:

1. In the administrative console, click **Servers > Application servers** and select the application server for which you want to automatically detect no-servant conditions.
2. Under Additional Properties, click **Custom Properties > New**.
3. Enter `control_region_dreg_on_no_srs` in the Name field and `1` in the Value field. When this custom property is set to 1, the server rejects all requests targeted for dispatch when it detects that there are no servants ready to process the requests. Setting this property to 0 (zero) turns off this function.
4. Enter `control_region_confirm_recovery_on_no_srs` in the Name field and either `0` (zero) or `1` in the Value field. If you enter `0` in the Value field, the controller resumes taking requests as soon as it detects that the minimal number of servants are ready to receive requests. If you enter `1` in the Value field, the controller issues WTOR message BBOO0297A as soon as it detects that the minimum

number of servants for which the server is configured are ready to accept work. The server waits until it receives a response to this message before it actually resumes taking requests.

5. Click **Review**, select **Synchronize changes with Nodes**, and then click **Save** to update the master repository with your changes.

Converting a 7-character server short name to 8 characters

By default, WebSphere Application Server for z/OS assumes you are using a 7-character short name (JOBNAME) for your application servers and deployment managers. If your naming standards require 8 characters, you can lengthen the 7-character server short name to 8 characters.

You should consider the following before performing this task:

- The Resource Recovery Services (RRS) log names are based on the server short name. When you change the server short name, you are changing the server's identity to the RRS. This means that the previously existing transaction and partner logs will be abandoned, or will not match the new name, and either of these situations will result in restart problems. To prevent this from happening, ensure that there are no outstanding RRS units of recovery (URs) for your server **before** changing its name. See *z/OS MVS Programming: Resource Recovery* for instructions on using the RRS panels to view information about URs.

  Note that the only safe way to provide an 8-character short name for a server is to do so before it is initially started.
- Converting your 7-character server short name to 8 characters requires you to change the JOBNAME used by the servant's start command. This means that the System Authorization Facility (SAF) started class that previously matched this job may no longer match. Review your SAF STARTED class profile and, if necessary, define a new class.
- Because the JOBNAME appears as part of a start command's arguments, you need to review your COMMNDxx PARMLIB member, as well as any other form of automation you use that issues a start command to start a WebSphere Application Server for z/OS server.
- Review the start parameters of your Workload Management (WLM) static APPLENV definitions. These are the parameters that are used to start the servant process (server region). If you are using static APPLENVs, the start parm string used by the WLM for your server looks similar to JOBNAME=BBOS001S,ENV=... You will need to decide if you want to keep this JOBNAME or change to the new JOBNAME that you specify in the steps below. The original JOBNAME should be sufficient.

  This consideration does not apply if you are using WLM dynamic APPLENVs.
- Review and update the necessary Resource Access Control Facility (RACF) profiles to support these server short names. See the *Securing applications and their environment* PDF for more information.

To lengthen a 7-character server short name to 8 characters:

1. Change the 7-character server short name to the 8-character name you wish to use:
   a. Navigate to the appropriate application server or deployment manager.

      For an application server, in the administrative console, click **Servers > Application Servers >** *server_name*.

      For a deployment manager, in the administrative console, click **System administration > Deployment manager**.

   b. In the **Short name** field, replace the 7-character name with the 8-character short name you wish to use.

   c. Click **OK**.
2. Update the start command arguments for the servant to use the new 8-character name. If you are reconfiguring a node agent, you can skip this step because it does not have an associated servant process.
   a. Navigate to the servant process.

For an application server, in the administrative console, click **Servers > Application Servers >** *server_name* , and then under Server Infrastructure, click **Java and Process Management > Process definition > Servant**.

For a deployment manager, in the administrative console, click **System administration > Deployment manager**, and then under Server Infrastructure, click **Java and process management > Process definition > Servant**.

b. In the startCommandArgs field, replace the 7-character name, designated by the JOBNAME argument, with the 8-character name you wish to use. Do not include the S character at the end of the JOBNAME. For example, JOBNAME=P5SVR1D,ENV=P5CELL.P5NODED.P5SVR1D

c. Click **OK**.

3. Click **Save** to save your configuration changes.

Core group service settings

Use this page to set up the application server properties that relate to core groups.

To view this administrative console page, click **Servers > Application servers>** > *server*. Then under Additional properties, select **Core group service**.

Click **Save** to save and synchronize your changes with all managed nodes.

Enable service at server startup

Select if you want the core group service, also known as the high availability manager service, to start on this process when the server starts. The core group service must be started before high availability functions, such as routing, and failover, work properly.

Important: Before disabling the core group service for a server process, see "When to use a high availability manager" on page 439 for a description of environments that might not require high availability functions.

Default	Core group service starts when the server starts.

Core group name

Specifies the name of the core group that contains this application server as a member. To move a server to a different core group, in the administrative console, click **Servers > Core groups > Core group settings >** *core_group* **> Core group servers**.

Data type	String

Allow activation

Select if high availability group members can be activated on this application server.

Is alive timer

Specifies the time interval, in seconds, at which the high availability manager will check the health of all of the active high availability group members that are running in this application server process. An active group member is a member that is able to accept work. If a group member fails, the application server on which the group member resides is restarted. If -1 is specified, the timer is disabled. If 0 (zero) is specified, the default value of 120 seconds is used.

Important: The value specified for this property can be overridden for the high availability groups using a particular policy if the Is alive timer property for that policy specifies a different time interval. If the Is alive timer setting specified for a policy is greater than 0 (zero), the high availability manager uses that time interval, instead of the one specified at this level, when determining how frequently it should check the health of a high availability group member using that particular policy.

Data type	Any integer between -1 and 600, inclusive
Default	120 seconds

Transport buffer size

Specifies the buffer size, in megabytes, of the underlying group communication transport. The minimum buffer size is 10 megabytes.

Data type	String
Default	10 megabytes

Setting the time zone for a single application server

You can ensure that your application components use the correct time zone. How you do this varies by the operating system on which WebSphere Application Server is installed and, in some cases, by the scope required.

In some application environments, it is important that application server components use the same time zone.

To change the time zone setting or a single application server, add the TZ Java virtual machine (JVM) custom property to the configuration settings of a specific application server:

1. In the administrative console click **Application Servers >** *server_name* .
2. Under Server Infrastructure, click **Java and Process Management> Process Definition** *process_name* **> Java Virtual Machine** > **Custom Properties**. For a base environment select the **Control** process. For a Network Deployment environment, select the **Servant** process.
3. Specify TZ in the Name field and the appropriate time zone in the Value field.

   Time zone names should include an offset and, in almost all cases, a daylight saving time zone name for consistent results. For example, you might specify EST5EDT for Eastern Standard Time, Daylight Savings Time.
4. **Optional:** Enter a description of the variable and the setting you specified.
5. Click **Apply**.
6. Save your work.

   Make sure Synchronize changes with Nodes is selected, and click **Save**.

All of the components of this application server use the time zone specified for the custom property.

Stop and restart this application server. You must restart the server for the change to take effect.

Supported time zone values

Use this page as a reference for time zone variables that are supported by WebSphere Application Server.

The following table lists the time zone values that WebSphere Application Server supports:

- The **Time zone ID** column lists time zones, in boldface, and the locations within each time zone.
- The **Raw offset** column lists the difference, in hours and minutes, between Greenwich Mean Time (GMT) and the specified time zone.
- The **DST offset** column lists the offset, in minutes, for Daylight Savings Time (DST). If the field is blank, the time zone does not use DST.
- The **Display name** column lists the names of the time zones.
- The **QTIMZON variable** column only applies to the i5/OS operating system. The **QTIMZON variable** column lists the corresponding value for the QTIMZON system variable. If multiple values are specified in this column, either value is acceptable.

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
Etc/GMT+12	-12 : 00		GMT-12:00	
Etc/GMT+11	-11 : 00		GMT-11:00	
MIT	-11 : 00		West Samoa Time	
Pacific/Apia	-11 : 00		West Samoa Time	QN1100UTCS
Pacific/Midway	-11 : 00		Samoa Standard Time	
Pacific/Niue	-11 : 00		Niue Time	
Pacific/Pago_Pago	-11 : 00		Samoa Standard Time	
Pacific/Samoa	-11 : 00		Samoa Standard Time	
US/Samoa	-11 : 00		Samoa Standard Time	
America/Adak	-10 : 00	60	Hawaii-Aleutian Standard Time	QN1000HAST
America/Atka	-10 : 00	60	Hawaii-Aleutian Standard Time	
Etc/GMT+10	-10 : 00		GMT-10:00	
HST	-10 : 00		Hawaii Standard Time	
Pacific/Fakaofo	-10 : 00		Tokelau Time	
Pacific/Honolulu	-10 : 00		Hawaii Standard Time	QN1000UTCS
Pacific/Johnston	-10 : 00		Hawaii Standard Time	
Pacific/Rarotonga	-10 : 00		Cook Is. Time	
Pacific/Tahiti	-10 : 00		Tahiti Time	
SystemV/HST10	-10 : 00		Hawaii Standard Time	
US/Aleutian	-10 : 00	60	Hawaii-Aleutian Standard Time	
US/Hawaii	-10 : 00		Hawaii Standard Time	
Pacific/Marquesas	-9 : 30		Marquesas Time	
AST	-9 : 00	60	Alaska Standard Time	QN0900AST
America/Anchorage	-9 : 00	60	Alaska Standard Time	
America/Juneau	-9 : 00	60	Alaska Standard Time	
America/Nome	-9 : 00	60	Alaska Standard Time	
America/Yakutat	-9 : 00	60	Alaska Standard Time	
Etc/GMT+9	-9 : 00		GMT-09:00	
Pacific/Gambier	-9 : 00		Gambier Time	QN0900UTCS
SystemV/YST9	-9 : 00	60	Alaska Standard Time	
US/Alaska	-9 : 00	60	Alaska Standard Time	
America/Dawson	-8 : 00	60	Pacific Standard Time	
America/Ensenada	-8 : 00	60	Pacific Standard Time	
America/Los_Angeles	-8 : 00	60	Pacific Standard Time	
America/Tiajuana	-8 : 00	60	Pacific Standard Time	
America/Vancouver	-8 : 00	60	Pacific Standard Time	
America/Whitehorse	-8 : 00	60	Pacific Standard Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
Canada/Pacific	-8 : 00	60	Pacific Standard Time	
Canada/Yukon	-8 : 00	60	Pacific Standard Time	
Etc/GMT+8	-8 : 00		GMT-08:00	
Mexico/BajaNorte	-8 : 00	60	Pacific Standard Time	
PST	-8 : 00	60	Pacific Standard Time	QN0800PST, QN0800U
PST8PDT	-8 : 00	60	Pacific Standard Time	
Pacific/Pitcairn	-8 : 00		Pitcairn Standard Time	QN0800UTCS
SystemV/PST8	-8 : 00		Pitcairn Standard Time	
SystemV/PST8PDT	-8 : 00	60	Pacific Standard Time	
US/Pacific	-8 : 00	60	Pacific Standard Time	
US/Pacific-New	-8 : 00	60	Pacific Standard Time	
America/Boise	-7 : 00	60	Mountain Standard Time	
America/Cambridge_Bay	-7 : 00	60	Mountain Standard Time	
America/Chihuahua	-7 : 00	60	Mountain Standard Time	
America/Dawson_Creek	-7 : 00		Mountain Standard Time	
America/Denver	-7 : 00	60	Mountain Standard Time	
America/Edmonton	-7 : 00	60	Mountain Standard Time	
America/Hermosillo	-7 : 00		Mountain Standard Time	
America/Inuvik	-7 : 00	60	Mountain Standard Time	
America/Mazatlan	-7 : 00	60	Mountain Standard Time	
America/Phoenix	-7 : 00		Mountain Standard Time	QN0700MST2, QN0700UTCS
America/Shiprock	-7 : 00	60	Mountain Standard Time	
America/Yellowknife	-7 : 00	60	Mountain Standard Time	
Canada/Mountain	-7 : 00	60	Mountain Standard Time	
Etc/GMT+7	-7 : 00		GMT-07:00	
MST	-7 : 00	60	Mountain Standard Time	QN0700MST, QN0700T
MST7MDT	-7 : 00	60	Mountain Standard Time	
Mexico/BajaSur	-7 : 00	60	Mountain Standard Time	
Navajo	-7 : 00	60	Mountain Standard Time	
PNT	-7 : 00	60	Mountain Standard Time	
SystemV/MST7	-7 : 00		Mountain Standard Time	
SystemV/MST7MDT	-7 : 00	60	Mountain Standard Time	
UA/Arizona	-7 : 00		Mountain Standard Time	
US/Mountain	-7 : 00	60	Mountain Standard Time	
America/Belize	-6 : 00		Central Standard Time	
America/Cancun	-6 : 00	60	Central Standard Time	
America/Chicago	-6 : 00	60	Central Standard Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
America/Costa_Rica	-6 : 00		Central Standard Time	QN0600UTCS
America/El_Salvador	-6 : 00		Central Standard Time	
America/Guatemala	-6 : 00		Central Standard Time	
America/Managua	-6 : 00		Central Standard Time	
America/Menominee	-6 : 00	60	Central Standard Time	
America/Merida	-6 : 00	60	Central Standard Time	
America/Mexico_City	-6 : 00	60	Central Standard Time	
America/Monterrey	-6 : 00	60	Central Standard Time	
America/North_Dakota/Center	-6 : 00	60	Central Standard Time	
America/Rainy_River	-6 : 00	60	Central Standard Time	
America/Rankin_Inlet	-6 : 00	60	Central Standard Time	
America/Regina	-6 : 00		Central Standard Time	
America/Swift_Current	-6 : 00		Central Standard Time	
America/Tegucigalpa	-6 : 00		Central Standard Time	
America/Winnipeg	-6 : 00	60	Central Standard Time	
CST	-6 : 00	60	Central Standard Time	QN0600CST, QN600S
CST6CDT	-6 : 00	60	Central Standard Time	
Canada/Central	-6 : 00	60	Central Standard Time	
Canada/East-Saskatchewan	-6 : 00		Central Standard Time	
Canada/Saskatchewan	-6 : 00		Central Standard Time	
Chile/EasterIsland	-6 : 00	60	Easter Is.Time	
Etc/GMT+6	-6 : 00		GMT-06:00	
Mexico/General	-6 : 00	60	Central Standard Time	
Pacific/Easter	-6 : 00	60	Easter Is. Time	
Pacific/Galapagos	-6 : 00		Galapagos Time	
Pacific/Easter	-6 : 00	60	Easter Is. Time	
Pacific/Galapagos	-6 : 00		Galapagos Time	
SystemV/CST6	-6 : 00		Central Standard Time	
SystemV/CST6CDT	-6 : 00	60	Central Standard Time	
US/Central	-6 : 00	60	Central Standard Time	
America/Bogota	-5 : 00		Colombia Time	
America/Cayman	-5 : 00		Eastern Standard Time	
America/Detroit	-5 : 00	60	Eastern Standard Time	
America/Eirunepe	-5 : 00		Acre Time	
America/Fort_Wayne	-5 : 00		Eastern Standard Time	
America/Grand_Turk	-5 : 00	60	Eastern Standard Time	
America/Guayaquil	-5 : 00		Ecuador Time	
America/Havana	-5 : 00	60	Central Standard Time	
America/Indiana/Indianapolis	-5 : 00		Eastern Standard Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
America/Indiana/Knox	-5 : 00		Eastern Standard Time	
America/Indiana/Marengo	-5 : 00		Eastern Standard Time	
America/Indiana/Vevay	-5 : 00		Eastern Standard Time	
America/Indianapolis	-5 : 00		Eastern Standard Time	QN0500UTCS
America/Iqaluit	-5 : 00	60	Eastern Standard Time	
America/Jamaica	-5 : 00		Eastern Standard Time	
America/Kentucky/Louisville	-5 : 00	60	Eastern Standard Time	
America/Kentucky/Monticello	-5 : 00	60	Eastern Standard Time	
America/Knox_IN	-5 : 00		Eastern Standard Time	
America/Lima	-5 : 00		Peru Time	
America/Louisville	-5 : 00	60	Eastern Standard Time	
America/Montreal	-5 : 00	60	Eastern Standard Time	
America/Nassau	-5 : 00	60	Eastern Standard Time	
America/New_York	-5 : 00	60	Eastern Standard Time	
America/Nipigon	-5 : 00	60	Eastern Standard Time	
America/Panama	-5 : 00		Eastern Standard Time	
America/Pangnirtung	-5 : 00	60	Eastern Standard Time	
America/Port-au-Prince	-5 : 00		Eastern Standard Time	
America/Porto_Acre	-5 : 00		Acre Time	
America/Rio_Branco	-5 : 00		Acre Time	
America/Thunder_Bay	-5 : 00	60	Eastern Standard Time	
Brazil/Acre	-5 : 00		Acre Time	
Canada/Eastern	-5 : 00	60	Eastern Standard Time	
Cuba	-5 : 00	60	Central Standard Time	
EST	-5 : 00	60	Eastern Standard Time	QN0500EST
EST5EDT	-5 : 00	60	Eastern Standard Time	
Etc/GMT+5	-5 : 00		GMT-05:00	
IET	-5 : 00		Eastern Standard Time	QN0500EST2
Jamaica	-5 : 00		Eastern Standard Time	
SystemV/EST5	-5 : 00		Eastern Standard Time	
SystemV/EST5EDT	-5 : 00	60	Eastern Standard Time	
US/East-Indiana	-5 : 00		Eastern Standard Time	
US/Eastern	-5 : 00	60	Eastern Standard Time	
US/Indiana-Starke	-5 : 00		Eastern Standard Time	
US/Michigan	-5 : 00	60	Eastern Standard Time	
America/Anguilla	-4 : 00		Atlantic Standard Time	
America/Antigua	-4 : 00		Atlantic Standard Time	
America/Aruba	-4 : 00		Atlantic Standard Time	
America/Asuncion	-4 : 00	60	Paraguay Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
America/Barbados	-4 : 00		Atlantic Standard Time	
America/Boa_Vista	-4 : 00		Amazon Standard Time	
America/Caracas	-4 : 00		Venezuela Time	QN0400UTC2
America/Cuiaba	-4 : 00	60	Amazon Standard Time	
America/Curacao	-4 : 00		Atlantic Standard Time	
America/Dominica	-4 : 00		Atlantic Standard Time	
America/Glace_Bay	-4 : 00	60	Atlantic Standard Time	
America/Goose_Bay	-4 : 00	60	Atlantic Standard Time	
America/Grenada	-4 : 00		Atlantic Standard Time	
America/Guadeloupe	-4 : 00		Atlantic Standard Time	
America/Guyana	-4 : 00		Guyana Time	
America/Halifax	-4 : 00	60	Atlantic Standard Time	
America/La_Paz	-4 : 00		Bolivia Time	
America/Manaus	-4 : 00		Amazon Standard Time	
America/Martinique	-4 : 00		Atlantic Standard Time	
America/Montserrat	-4 : 00		Atlantic Standard Time	
America/Port_of_Spain	-4 : 00		Atlantic Standard Time	
America/Porto_Velho	-4 : 00		Amazon Standard Time	
America/Puerto_Rico	-4 : 00		Atlantic Standard Time	QN0400UTCS
America/Santiago	-4 : 00	60	Chile Time	
America/Santo_Domingo	-4 : 00		Atlantic Standard Time	
America/St_Kitts	-4 : 00		Atlantic Standard Time	
America/St_Lucia	-4 : 00		Atlantic Standard Time	
America/St_Thomas	-4 : 00		Atlantic Standard Time	
America/St_Vincent	-4 : 00		Atlantic Standard Time	
America/Thule	-4 : 00	60	Atlantic Standard Time	
America/Tortola	-4 : 00		Atlantic Standard Time	
America/Virgin	-4 : 00		Atlantic Standard Time	
Antarctica/Palmer	-4 : 00	60	Chile Time	
Atlantic/Bermuda	-4 : 00	60	Atlantic Standard Time	QN0400AST
Atlantic/Stanley	-4 : 00	60	Falkland Is. Time	
Brazil/West	-4 : 00		Amazon Standard Time	
Canada/Atlantic	-4 : 00	60	Atlantic Standard Time	
Chile/Continental	-4 : 00	60	Chile Time	
Etc/GMT+4	-4 : 00		GMT-04:00	
PRT	-4 : 00		Atlantic Standard Time	
SystemV/AST4	-4 : 00		Atlantic Standard Time	
SystemV/AST4ADT	-4 : 00	60	Atlantic Standard Time	
America/St_Johns	-3 : 30	60	Newfoundland Standard Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
CNT	-3 : 30	60	Newfoundland Standard Time	QN0330NST
Canada/Newfoundland	-3 : 30	60	Newfoundland Standard Time	
AGT	-3 : 00		Argentine Time	
America/Araguaina	-3 : 00	60	Brazil Time	
America/Belem	-3 : 00		Brazil Time	
America/Buenos_Aires	-3 : 00		Argentine Time	QN0300UTCS
America/Catamarca	-3 : 00		Argentine Time	
America/Cayenne	-3 : 00		French Guiana Time	
America/Cordoba	-3 : 00		Argentine Time	
America/Fortaleza	-3 : 00		Brazil Time	
America/Godthab	-3 : 00	60	Western Greenland Time	
America/Jujuy	-3 : 00		Argentine Time	
America/Maceio	-3 : 00		Brazil Time	
America/Mendoza	-3 : 00		Argentine Time	
America/Miquelon	-3 : 00	60	Pierre & Miquelon Standard Time	
America/Montevideo	-3 : 00		Uruguay Time	
America/Paramaribo	-3 : 00		Suriname Time	
America/Recife	-3 : 00		Brazil Time	
America/Rosario	-3 : 00		Argentine Time	
America/Sao_Paulo	-3 : 00	60	Brazil Time	
Antarctica/Rothera	-3 : 00		Rothera Time	
BET	-3 : 00	60	Brazil Time	QN0300UTC2
Brazil/East	-3 : 00	60	Brazil Time	
Etc/GMT+3	-3 : 00		GMT-03:00	
America/Noronha	-2 : 00		Fernando de Noronha Time	QN0200UTCS
Atlantic/South_Georgia	-2 : 00		South Georgia Standard Time	
Brazil/DeNoronha	-2 : 00		Fernando de Noronha Time	
Etc/GMT+2	-2 : 00		GMT-02:00	
America/Scoresbysund	-1 : 00	60	Eastern Greenland Time	
Atlantic/Azores	-1 : 00	60	Azores Time	
Atlantic/Cape_Verde	-1 : 00		Cape Verde Time	QN0100UTCS
Etc/GMT+1	-1 : 00		GMT-01:00	
Africa/Abidjan	0 : 00		Greenwich Mean Time	
Africa/Accra	0 : 00		Greenwich Mean Time	
Africa/Bamako	0 : 00		Greenwich Mean Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
Africa/Banjul	0 : 00		Greenwich Mean Time	
Africa/Bissau	0 : 00		Greenwich Mean Time	
Africa/Casablanca	0 : 00		Western European Time	
Africa/Conakry	0 : 00		Greenwich Mean Time	
Africa/Dakar	0 : 00		Greenwich Mean Time	
Africa/El_Aaiun	0 : 00		Western European Time	
Africa/Freetown	0 : 00		Greenwich Mean Time	
Africa/Lome	0 : 00		Greenwich Mean Time	
Africa/Monrovia	0 : 00		Greenwich Mean Time	
Africa/Nouakchott	0 : 00		Greenwich Mean Time	
Africa/Ouagadougou	0 : 00		Greenwich Mean Time	
Africa/Sao_Tome	0 : 00		Greenwich Mean Time	
Africa/Timbuktu	0 : 00		Greenwich Mean Time	
America/Danmarkshavn	0 : 00		Greenwich Mean Time	
Atlantic/Canary	0 : 00	60	Western European Time	
Atlantic/Faeroe	0 : 00	60	Western European Time	
Atlantic/Madeira	0 : 00	60	Western European Time	
Atlantic/Reykjavik	0 : 00		Greenwich Mean Time	
Atlantic/St_Helena	0 : 00		Greenwich Mean Time	
Eire	0 : 00	60	Greenwich Mean Time	
Etc/GMT	0 : 00		GMT+00:00	
Etc/GMT+0	0 : 00		GMT+00:00	
Etc/GMT-0	0 : 00		GMT+00:00	
Etc/GMT0	0 : 00		GMT+00:00	
Etc/Greenwich	0 : 00		Greenwich Mean Time	
Etc/UCT	0 : 00		Coordinated Universal Time	
Etc/UTC	0 : 00		Coordinated Universal Time	
Etc/Universal	0 : 00		Coordinated Universal Time	
Etc/Zulu	0 : 00		Coordinated Universal Time	
Europe/Belfast	0 : 00	60	Greenwich Mean Time	
Europe/Dublin	0 : 00	60	Greenwich Mean Time	
Europe/Lisbon	0 : 00	60	Western European Time	
Europe/London	0 : 00	60	Greenwich Mean Time	Q0000GMT2
GB	0 : 00	60	Greenwich Mean Time	
GB-Eire	0 : 00	60	Greenwich Mean Time	
GMT	0 : 00		Greenwich Mean Time	Q0000GMT

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
GMT0	0 : 00		GMT+00:00	
Greenwich	0 : 00		Greenwich Mean Time	
Iceland	0 : 00		Greenwich Mean Time	
Portugal	0 : 00	60	Western European Time	
UCT	0 : 00		Coordinated Universal Time	
UTC	0 : 00		Coordinated Universal Time	Q0000UTC
Universal	0 : 00		Coordinated Universal Time	
WET	0 : 00	60	Western European Time	
Zulu	0 : 00		Coordinated Universal Time	
Africa/Algiers	1 : 00		Central European Time	QP0100CET, QP0100UTCS
Africa/Bangui	1 : 00		Western African Time	
Africa/Brazzaville	1 : 00		Western African Time	
Africa/Ceuta	1 : 00	60	Central European Time	
Africa/Douala	1 : 00		Western African Time	
Africa/Kinshasa	1 : 00		Western African Time	
Africa/Lagos	1 : 00		Western African Time	
Africa/Libreville	1 : 00		Western African Time	
Africa/Luanda	1 : 00		Western African Time	
Africa/Malabo	1 : 00		Western African Time	
Africa/Ndjamena	1 : 00		Western African Time	
Africa/Niamey	1 : 00		Western African Time	
Africa/Porto-Novo	1 : 00		Western African Time	
Africa/Tunis	1 : 00		Central European Time	
Africa/Windhoek	1 : 00	60	Western African Time	
Arctic/Longyearbyen	1 : 00	60	Central European Time	
Atlantic/Jan_Mayen	1 : 00	60	Eastern Greenland Time	
CET	1 : 00	60	Central European Time	
ECT	1 : 00	60	Central European Time	QP0100CET3
Etc/GMT-1	1 : 00		GMT+01:00	
Europe/Amsterdam	1 : 00	60	Central European Time	
Europe/Andorra	1 : 00	60	Central European Time	
Europe/Belgrade	1 : 00	60	Central European Time	
Europe/Berlin	1 : 00	60	Central European Time	
Europe/Bratislava	1 : 00	60	Central European Time	
Europe/Brussels	1 : 00	60	Central European Time	
Europe/Budapest	1 : 00	60	Central European Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
Europe/Copenhagen	1 : 00	60	Central European Time	
Europe/Gibraltar	1 : 00	60	Central European Time	
Europe/Ljubljana	1 : 00	60	Central European Time	
Europe/Luxembourg	1 : 00	60	Central European Time	
Europe/Madrid	1 : 00	60	Central European Time	
Europe/Malta	1 : 00	60	Central European Time	
Europe/Monaco	1 : 00	60	Central European Time	
Europe/Oslo	1 : 00	60	Central European Time	
Europe/Paris	1 : 00	60	Central European Time	
Europe/Prague	1 : 00	60	Central European Time	
Europe/Rome	1 : 00	60	Central European Time	
Europe/San_Marino	1 : 00	60	Central European Time	
Europe/Sarajevo	1 : 00	60	Central European Time	
Europe/Skopje	1 : 00	60	Central European Time	
Europe/Stockholm	1 : 00	60	Central European Time	
Europe/Tirane	1 : 00	60	Central European Time	
Europe/Vaduz	1 : 00	60	Central European Time	
Europe/Vatican	1 : 00	60	Central European Time	
Europe/Vienna	1 : 00	60	Central European Time	
Europe/Warsaw	1 : 00	60	Central European Time	
Europe/Zagreb	1 : 00	60	Central European Time	
Europe/Zurich	1 : 00	60	Central European Time	QP0100CET2
MET	1 : 00	60	Middle Europe Time	
Poland	1 : 00	60	Central European Time	
ART	2 : 00	60	Eastern European Time	
Africa/Blantyre	2 : 00		Central African Time	
Africa/Bujumbura	2 : 00		Central African Time	
Africa/Cairo	2 : 00	60	Eastern European Time	
Africa/Gaborone	2 : 00		Central African Time	
Africa/Harare	2 : 00		Central African Time	
Africa/Johannesburg	2 : 00		South Africa Standard Time	QP0200SAST
Africa/Kigali	2 : 00		Central African Time	
Africa/Lubumbashi	2 : 00		Central African Time	
Africa/Lusaka	2 : 00		Central African Time	
Africa/Maputo	2 : 00		Central African Time	
Africa/Maseru	2 : 00		South Africa Standard Time	
Africa/Mbabane	2 : 00		South Africa Standard Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
Africa/Tripoli	2 : 00		Eastern European Time	
Asia/Amman	2 : 00	60	Eastern European Time	
Asia/Beirut	2 : 00	60	Eastern European Time	
Asia/Damascus	2 : 00	60	Eastern European Time	
Asia/Gaza	2 : 00	60	Eastern European Time	
Asia/Istanbul	2 : 00	60	Eastern European Time	
Asia/Jerusalem	2 : 00	60	Israel Standard Time	
Asia/Nicosia	2 : 00	60	Eastern European Time	
Asia/Tel_Aviv	2 : 00	60	Israel Standard Time	
CAT	2 : 00		Central African Time	
EET	2 : 00	60	Eastern European Time	QP0200EET
Egypt	2 : 00	60	Eastern European Time	
Etc/GMT-2	2 : 00		GMT+02:00	
Europe/Athens	2 : 00	60	Eastern European Time	
Europe/Bucharest	2 : 00	60	Eastern European Time	
Europe/Chisinau	2 : 00	60	Eastern European Time	
Europe/Helsinki	2 : 00	60	Eastern European Time	
Europe/Istanbul	2 : 00	60	Eastern European Time	
Europe/Kaliningrad	2 : 00	60	Eastern European Time	
Europe/Kiev	2 : 00	60	Eastern European Time	
Europe/Minsk	2 : 00	60	Eastern European Time	
Europe/Nicosia	2 : 00	60	Eastern European Time	
Europe/Riga	2 : 00	60	Eastern European Time	
Europe/Simferopol	2 : 00	60	Eastern European Time	
Europe/Sofia	2 : 00	60	Eastern European Time	
Europe/Tallinn	2 : 00	60	Eastern European Time	QP0200EET2, QP0200UTCS
Europe/Tiraspol	2 : 00	60	Eastern European Time	
Europe/Uzhgorod	2 : 00	60	Eastern European Time	
Europe/Vilnius	2 : 00	60	Eastern European Time	
Europe/Zaporozhye	2 : 00	60	Eastern European Time	
Israel	2 : 00	60	Israel Standard Time	
Libya	2 : 00		Eastern European Time	
Turkey	2 : 00	60	Eastern European Time	
Africa/Addis_Ababa	3 : 00		Eastern African Time	QP0300UTCS
Africa/Asmera	3 : 00		Eastern African Time	
Africa/Dar_es_Salaam	3 : 00		Eastern African Time	
Africa/Djibouti	3 : 00		Eastern African Time	
Africa/Kampala	3 : 00		Eastern African Time	
Africa/Khartoum	3 : 00		Eastern African Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
Africa/Mogadishu	3 : 00		Eastern African Time	
Africa/Nairobi	3 : 00		Eastern African Time	
Antarctica/Syowa	3 : 00		Syowa Time	
Asia/Aden	3 : 00		Arabia Standard Time	
Asia/Baghdad	3 : 00	60	Arabia Standard Time	
Asia/Bahrain	3 : 00		Arabia Standard Time	
Asia/Kuwait	3 : 00		Arabia Standard Time	
Asia/Qatar	3 : 00		Arabia Standard Time	
Asia/Riyadh	3 : 00		Arabia Standard Time	
EAT	3 : 00		Eastern African Time	
Etc/GMT-3	3 : 00		GMT+03:00	
Europe/Moscow	3 : 00	60	Moscow Standard Time	
Indian/Antananarivo	3 : 00		Eastern African Time	
Indian/Comoro	3 : 00		Eastern African Time	
Indian/Mayotte	3 : 00		Eastern African Time	
W-SU	3 : 00	60	Moscow Standard Time	
Asia/Riyadh87	3 : 07		GMT+03:07	
Asia/Riyadh88	3 : 07		GMT+03:07	
Asia/Riyadh89	3 : 07		GMT+03:07	
Mideast/Riyadh87	3 : 07		GMT+03:07	
Mideast/Riyadh88	3 : 07		GMT+03:07	
Mideast/Riyadh89	3 : 07		GMT+03:07	
Asia/Tehran	3 : 30	60	Iran Standard Time	
Iran	3 : 30	60	Iran Standard Time	
Asia/Aqtau	4 : 00	60	Aqtau Time	QP0400UTC2
Asia/Baku	4 : 00	60	Azerbaijan Time	
Asia/Dubai	4 : 00		Gulf Standard Time	QP0400UTCS
Asia/Muscat	4 : 00		Gulf Standard Time	
Asia/Oral	4 : 00	60	Oral Time	
Asia/Tbilisi	4 : 00	60	Georgia Time	
Asia/Yerevan	4 : 00	60	Armenia Time	
Etc/GMT-4	4 : 00		GMT+04:00	
Europe/Samara	4 : 00	60	Samara Time	
Indian/Mahe	4 : 00		Seychelles Time	
Indian/Mauritius	4 : 00		Mauritius Time	
Indian/Reunion	4 : 00		Reunion Time	
NET	4 : 00	60	Armenia Time	
Asia/Kabul	4 : 30		Afghanistan Time	
Asia/Aqtobe	5 : 00	60	Aqtobe Time	QP0500UTC2

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
Asia/Ashgabat	5 : 00		Turkmenistan Time	
Asia/Ashkhabad	5 : 00		Turkmenistan Time	
Asia/Bishkek	5 : 00	60	Kirgizstan Time	
Asia/Dushanbe	5 : 00		Tajikistan Time	
Asia/Karachi	5 : 00		Pakistan Time	QP0500UTCS
Asia/Samarkand	5 : 00		Turkmenistan Time	
Asia/Tashkent	5 : 00		Uzbekistan Time	
Asia/Yekaterinburg	5 : 00	60	Yekaterinburg Time	
Etc/GMT-5	5 : 00		GMT+05:00	
Indian/Kerguelen	5 : 00		French Southern & Antarctic Lands Time	
Indian/Maldives	5 : 00		Maldives Time	
PLT	5 : 00		Pakistan Time	
Asia/Calcutta	5 : 30		India Standard Time	
IST	5 : 30		India Standard Time	QP0530IST
Asia/Katmandu	5 : 45		Nepal Time	
Antarctica/Mawson	6 : 00		Mawson Time	
Antarctica/Vostok	6 : 00		Vostok Time	
Asia/Almaty	6 : 00	60	Alma-Ata Time	QP0600UTC2
Asia/Colombo	6 : 00		Sri Lanka Time	
Asia/Dacca	6 : 00		Bangladesh Time	
Asia/Dhaka	6 : 00		Bangladesh Time	QP0600UTCS
Asia/Novosibirsk	6 : 00	60	Novosibirsk Time	
Asia/Omsk	6 : 00	60	Omsk Time	
Asia/Qyzylorda	6 : 00	60	Qyzylorda Time	
Asia/Thimbu	6 : 00		Bhutan Time	
Asia/Thimphu	6 : 00		Bhutan Time	
BST	6 : 00		Bangladesh Time	
Etc/GMT-6	6 : 00		GMT+06:00	
Indian/Chagos	6 : 00		Indian Ocean Territory Time	
Asia/Rangoon	6 : 30		Myanmar Time	
Indian/Cocos	6 : 30		Cocos Islands Time	
Antarctica/Davis	7 : 00		Davis Time	
Asia/Bangkok	7 : 00		Indochina Time	
Asia/Hovd	7 : 00		Hovd Time	
Asia/Jakarta	7 : 00		West Indonesia Time	QP0700WIB
Asia/Krasnoyarsk	7 : 00	60	Krasnoyarsk Time	
Asia/Phnom_Penh	7 : 00		Indochina Time	
Asia/Pontianak	7 : 00		West Indonesia Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
Asia/Saigon	7 : 00		Indochina Time	QP0700UTCS
Asia/Vientiane	7 : 00		Indochina Time	
Etc/GMT-7	7 : 00		GMT+07:00	
Indian/Christmas	7 : 00		Christmas Island Time	
VST	7 : 00		Indochina Time	
Antarctica/Casey	8 : 00		Western Standard Time (Australia)	
Asia/Brunei	8 : 00		Brunei Time	
Asia/Chongqing	8 : 00		China Standard Time	
Asia/Chungking	8 : 00		China Standard Time	
Asia/Harbin	8 : 00		China Standard Time	
Asia/Hong_Kong	8 : 00		Hong Kong Time	QP0800JIST, QP0800UTCS
Asia/Irkutsk	8 : 00	60	Irkutsk Time	
Asia/Kashgar	8 : 00		China Standard Time	
Asia/Kuala_Lumpur	8 : 00		Malaysia Time	
Asia/Kuching	8 : 00		Malaysia Time	
Asia/Macao	8 : 00		China Standard Time	
Asia/Macau	8 : 00		China Standard Time	
Asia/Makassar	8 : 00		Central Indonesia Time	
Asia/Manila	8 : 00		Philippines Time	
Asia/Shanghai	8 : 00		China Standard Time	
Asia/Singapore	8 : 00		Singapore Time	
Asia/Taipei	8 : 00		China Standard Time	
Asia/Ujung_Pandang	8 : 00		Central Indonesia Time	QP0800WITA
Asia/Ulaanbaatar	8 : 00		Ulaanbaatar Time	
Asia/Ulan_Bator	8 : 00		Ulaanbaatar Time	
Asia/Urumqi	8 : 00		China Standard Time	
Australia/Perth	8 : 00		Western Standard Time (Australia)	QP0800AWST
Australia/West	8 : 00		Western Standard Time (Australia)	
CTT	8 : 00		China Standard Time	QP0800BST
Etc/GMT-8	8 : 00		GMT+08:00	
Hongkong	8 : 00		Hong Kong Time	
PRC	8 : 00		China Standard Time	
Singapore	8 : 00		Singapore Time	
Asia/Choibalsan	9 : 00		Choibalsan Time	
Asia/Dili	9 : 00		East Timor Time	
Asia/Jayapura	9 : 00		East Indonesia Time	QP0900WIT

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
Asia/Pyongyang	9 : 00		Korea Standard Time	
Asia/Seoul	9 : 00		Korea Standard Time	QP0900KST
Asia/Tokyo	9 : 00		Japan Standard Time	QP0900UTCS
Asia/Yakutsk	9 : 00	60	Yakutsk Time	
Etc/GMT-9	9 : 00		GMT+09:00	
JST	9 : 00		Japan Standard Time	QP0900JST
Japan	9 : 00		Japan Standard Time	
Pacific/Palau	9 : 00		Palau Time	
ROK	9 : 00		Korea Standard Time	
ACT	9 : 30		Central Standard Time (Northern Territory)	
Australia/Adelaide	9 : 30	60	Central Standard Time (South Australia)	QP0930ACST
Australia/Broken_Hill	9 : 30	60	Central Standard Time (South Australia/New South Wales)	
Australia/Darwin	9 : 30		Central Standard Time (Northern Territory)	
Australia/North	9 : 30		Central Standard Time (Northern Territory)	
Australia/South	9 : 30	60	Central Standard Time (South Australia)	
Australia/Yancowinna	9 : 30	60	Central Standard Time (South Australia/New South Wales)	
AET	10 : 00	60	Eastern Standard Time (New South Wales)	QP1000AEST
Antarctica/DumontDUrville	10 : 00		Dumont-d'Urville Time	
Asia/Sakhalin	10 : 00	60	Sakhalin Time	
Asia/Vladivostok	10 : 00	60	Vladivostok Time	
Australia/ACT	10 : 00	60	Eastern Standard Time (New South Wales)	
Australia/Brisbane	10 : 00		Eastern Standard Time (Queensland)	
Australia/Canberra	10 : 00	60	Eastern Standard Time (New South Wales)	
Australia/Hobart	10 : 00	60	Eastern Standard Time (Tasmania)	
Australia/Lindeman	10 : 00		Eastern Standard Time (Queensland)	
Australia/Melbourne	10 : 00	60	Eastern Standard Time (Victoria)	
Australia/NSW	10 : 00	60	Eastern Standard Time (New South Wales)	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
Australia/Queensland	10 : 00		Eastern Standard Time (Queensland)	
Australia/Sydney	10 : 00	60	Eastern Standard Time (New South Wales)	
Australia/Tasmania	10 : 00	60	Eastern Standard Time (Tasmania)	
Australia/Victoria	10 : 00	60	Eastern Standard Time (Victoria)	
Etc/GMT-10	10 : 00		GMT+10:00	
Pacific/Guam	10 : 00		Chamorro Standard Time	QP1000UTCS
Pacific/Port_Moresby	10 : 00		Papua New Guinea Time	
Pacific/Saipan	10 : 00		Chamorro Standard Time	
Pacific/Truk	10 : 00		Truk Time	
Pacific/Yap	10 : 00		Yap Time	
Australia/LHI	10 : 30	30	Load Howe Standard Time	
Australia/Lord_Howe	10 : 30	30	Load Howe Standard Time	
Asia/Magadan	11 : 00	60	Magadan Time	
Etc/GMT-11	11 : 00		GMT+11:00	
Pacific/Efate	11 : 00		Vanuatu Time	
Pacific/Guadalcanal	11 : 00		Solomon Is. Time	QP1100UTCS
Pacific/Kosrae	11 : 00		Kosrae Time	
Pacific/Noumea	11 : 00		New Caledonia Time	
Pacific/Ponape	11 : 00		Ponape Time	
SST	11 : 00		Solomon Is. Time	
Pacific/Norfolk	11 : 30		Norfolk Time	
Antarctica/McMurdo	12 : 00	60	New Zealand Standard Time	
Antarctica/South_Pole	12 : 00	60	New Zealand Standard Time	
Asia/Anadyr	12 : 00	60	Anadyr Time	
Asia/Kamchatka	12 : 00	60	Petropavlovsk-Kamchatski Time	
Etc/GMT-12	12 : 00		GMT+12:00	
Kwajalein	12 : 00		Marshall Islands Time	
NST	12 : 00	60	New Zealand Standard Time	QP1200NZST
NZ	12 : 00	60	New Zealand Standard Time	
Pacific/Auckland	12 : 00	60	New Zealand Standard Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
Pacific/Fiji	12 : 00		Fiji Time	QN1200UTCS, QP1200UTCS
Pacific/Funafuti	12 : 00		Tuvalu Time	
Pacific/Kwajalein	12 : 00		Marshall Islands Time	
Pacific/Majuro	12 : 00		Marshall Islands Time	
Pacific/Nauru	12 : 00		Nauru Time	
Pacific/Tarawa	12 : 00		Gilbert Is. Time	
Pacific/Wake	12 : 00		Wake Time	
Pacific/Wallis	12 : 00		Wallis & Futuna Time	
NZ-CHAT	12 : 45	60	Chatham Standard Time	
Pacific/Chatham	12 : 45	60	Chatham Standard Time	QP1245UTCS
Etc/GMT-13	13 : 00		GMT+13:00	
Pacific/Enderbury	13 : 00		Phoenix Is. Time	
Pacific/Tongatapu	13 : 00		Tonga Time	
Etc/GMT-14	14 : 00		GMT+14:00	
Pacific/Kiritimati	14 : 00		Line Is. Time	

Web module or application server stops processing requests

Use this information to help determine why a Web module or application server has stopped processing new requests.

If an application server's process spontaneously closes, or its Web modules stop responding to new requests:

- Isolate the problem by installing Web modules on different servers, if possible.
- You can use the Tivoli performance viewer to determine which resources have reached their maximum capacity, such as Java heap memory (indicating a possible memory leak) and database connections. If a particular resource appears to have reached its maximum capacity, review the application code for a possible cause:
  - If database connections are used and never freed, ensure that application code performs a **close()** on any opened **Connection** object within a **finally{}** block.
  - If there is a steady increase in servlet engine threads in use, review application **synchronized** code blocks for possible deadlock conditions.
  - If there is a steady increase in a JVM heap size, review application code for memory leak opportunities, such as static (class-level) collections, that can cause objects to never get garbage-collected.

  See the *Tuning guide* PDF for more information about this tool.
- To detect memory leak problems, enable verbose garbage collection on the application server. This feature adds detailed statements to the JVM error log file of the application server about the amount of available and in-use memory. To set up verbose garbage collection:
  1. Select **Servers > Application Servers >** *server_name* **> Java and Process Management > Process Definition > Java Virtual Machine**, and enable **Verbose Garbage Collection**.
  2. Stop and restart the application server.
  3. Periodically, or after the application server stops, browse the log file for garbage collection statements. Look for statements beginning with ″allocation failure″. The string indicates that a need for memory allocation has triggered a JVM garbage collection (freeing of unused memory).

Allocation failures themselves are normal and not necessarily indicative of a problem. The allocation failure statement is followed by statements showing how many bytes are needed and how many are allocated.

If there is a steady increase in the total amount of free and used memory (the JVM keeps allocating more memory for itself), or if the JVM becomes unable to allocate as much memory as it needs (indicated by the bytes needed statement), there might be a memory leak.

- If an application server's process spontaneously closes, there will be an SDUMP. See the *Troubleshooting and support* PDF for instructions on how to analyze the dump.

IBM Support has documents and tools that can save you time gathering information needed to resolve problems as described in Troubleshooting help from IBM. Before opening a problem report, see the Support page:

- http://www.ibm.com/software/webservers/appserv/zos_os390/support/

Creating generic servers

A generic server is a server that is managed in the WebSphere administrative domain, although it is not a server that is supplied by WebSphere Application Server. The WebSphere Application Server generic servers function enables you to define a generic server as an application server instance within the WebSphere Application Server administration, and associate it with a non-WebSphere server or process.

Generic application servers must be non-Java application processes that are either a started task or a shell script. You cannot create a Java application as a generic server for WebSphere Application Server on the z/OS platform.

The following processes can be created as a generic server provided that they are either started tasks or a shell scripts:

- A C or C++ server or process
- A CORBA server
- A Remote Method Invocation (RMI) server

You can use the wsadmin tool or the administrative console to create a generic server.

- **Create a non-Java application as a generic server.** The following steps describe how to use the administrative console to create a non-Java application as a generic application server.
  1. Select **Servers** > **Generic servers**
  2. Click **New**.
  3. Type in a name for the generic server.

     The name must be unique within the node. It is recommended that you use a naming scheme that makes it easy to distinguish your generic application servers from regular WebSphere Application Servers.
  4. Select a template for the new server. You can use a default application server template for your new server or use an existing application server as a template. The new application server will inherit all properties of the template server. If you create the new server using an existing application server do not enable the option to map applications from the existing server to the new server. This option does not apply for a generic server.
  5. Click **Next**
  6. Click **Finish**. The generic server now appears as an option on the **Generic servers** page in the administrative console.
  7. On the **Generic servers** page, click on the name of the generic server.
  8. Under Additional Properties, click **Process Definition**.
  9. In the Executable name field under General Properties, enter the name of the non-WebSphere Application Server program that is launched when you start this generic server. Executable target

type and Executable target properties are not used for non-Java applications. Executable target type and Executable target properties are only used for Java applications

10. Click **OK**.

- **Create a Java application as a generic server:** The following steps describe how to use the administrative console to create a Java application as a generic application server.
  1. Select **Servers** > **Generic servers**
  2. Click **New**.
  3. Type in a name for the generic server.

     The name must be unique within the node. It is highly recommended that you use a naming scheme that makes it easy to distinguish your generic application servers from regular WebSphere Application Servers. This name scheme enables you to quickly determine whether to use the **Terminate** or **Stop** button in the administrative console to stop specific application server. You must use the **Terminate** button to stop a generic application server.
  4. Click **Next**
  5. Click **Finish**. The generic server now appears as an option on the **Applications Server** page in the administrative console.
  6. Click **Finish**. The generic server now appears as an option on the **Generic servers** page in the administrative console.
  7. On the Generic servers page, click on the name of the generic server.
  8. Under Additional Properties, click **Process Definition**.
  9. In the Executable name field under General Properties, enter the path for the WebSphere Application Server default JVM, ${JAVA_HOME}/bin/java, which is used to run the Java application when you start this generic server.
  10. In the Executable target type field under General Properties, select whether a Java class name, **JAVA_CLASS**, or the name of an executable JAR file, **EXECUTABLE_JAR**, is used as the executable target of this Java process. The default for WebSphere Application Server is **JAVA_CLASS**.
  11. In the Executable target field under General Properties, enter the name of the executable target. Depending on the executable target type, this is either a Java class containing a main() method, or the name of an executable JAR file.) The default for WebSphere Application Server is com.ibm.ws.runtime.WsServer.
  12. Click **OK**.

      Note: If the generic server is to run an application server other than the WebSphere Application Server, leave the Executable name field set to the default value and specify the Java class containing the main function for your application serve in the Executable target field.

After you define a generic server, use the Application Server administrative console to start, stop, and monitor the associated non-WebSphere server or process when stopping or starting the applications that rely on them.

Important: You can use either the **Terminate** or **Stop** buttons in the administrative console to stop any application server, including a generic application server.

Starting and terminating generic application servers

This topic describes how to start and terminate generic servers.

If you create a generic server on a base WebSphere Application Server, you cannot use the base Application Server administrative console to start or terminate this server. You must use the wsadmin tool to manage this server.

If you create a generic server in a Network Deployment environment, you can use the administrative console to start and terminate this server.

1. Start a generic application server.

   There are two ways to start a generic server in a Network Deployment environment. You can use the MBean NodeAgent launchProcess operation of the wsadmin tool, or you can use the administrative console. To use the administrative console:

   a. In the administrative console, click **Servers > Application Servers**.

   b. Select the name of the generic server you want to start, and then click **Start**.

   a. View the **Status** value and any messages or logs to see whether the generic server starts.

2. Terminate generic servers.

   There are two ways to terminate a generic server in a Network Deployment environment. You can use the MBean terminate launchProcess operation of the wsadmin tool or you can use the administrative console. To use the administrative console:

   a. In the administrative console, click **Servers > Application Servers**.

   b. Select the check box beside the name of the generic server, and then click **Terminate**.

      Restriction: The **Stop** and **Stop Immediate** buttons on the administrative console do not work for generic servers.

   c. View the **Status** value and any messages or logs to see whether the generic server terminates.

Setting up peer restart and recovery

To allow WebSphere Application Server for z/OS to restart on an alternate system, the following prerequisites must be installed on every system (your original system as well as any systems intended for recovery) before reconfiguring the ARM policies to enable peer restart and recovery.

Important: Peer Restart and Recovery (PRR) functionality is deprecated. You should use the integrated high availability support for the transaction service subcomponent, instead of Peer Restart and Recovery for transaction recovery. See Transactional high availability and Configuring transaction properties for peer recovery for more information about the integrated high availability support for the transaction service subcomponent and how to configure it for peer recovery of transactions being processed on a application server that fails.

You must also make sure all of the systems, where you might need to perform restart, are part of the same RRS log group.

- z/OS Version 1.2 or higher
- BCP APAR OA01584
- RRS APARs OA02556 and OA2556
- WebSphere Application Server for z/OS Version 5 or higher

Installing the prerequisite service updates on all of these systems will not hinder your current running environment if you want to continue to only restart in place. However, if this service is not installed, there is a possibility that the controller will not be able to move back. OTS will attempt to restart on the alternate system and fail. If there are any URs that are unresolved with RRS once this happens, the controller will not be allowed to restart on the home system until RRS is cancelled on the alternate system. For more information on OTS and RRS, see *z/OS MVS Programming: Resource Recovery*.

If you do not plan to use peer restart and recovery, you do not need to abide by these functional prerequisites. Your system will instead use the restart-in-place function.

The following products all support RRS. Individually, they also support peer restart and recovery, providing the above prerequisites are all properly installed:

- DB2 Version 7 or higher

- IMS Version 8 or higher
- CICS Version 1.3 or higher
- MQSeries Version 5.2 or higher

In addition to the preceding products, many JTA XAResource Managers can be used to assist in a WebSphere Application Server for z/OS peer restart and recovery. Consult your JTA XAResource Manager's documentation to determine if it supports restarting on an alternate system.

Important: When setting up the ARM policy for a sysplex, make sure that both systems have the same level of the Application Server installed. For example, you cannot use an application server that is running WebSphere Application Server for z/OS Version 5.1 to perform peer restart and recovery for an application server that is running WebSphere Application Server for z/OS Version 6.0.1.

Prior to using peer restart and recovery:

- You must ensure that the location service Daemon and node agent are already running on all systems that might be used for recovery. Otherwise, the recovering system might attempt to recover on a system that is not running the location service Daemon and node agent. If this happens, the server will fail to start, and recovery will fail.

Clients will see a performance impact if the systems are running at capacity. In an attempt to minimize the memory and CPU impact on the alternate system, the enterprise bean and Web containers are not restarted for servers running in peer-restart mode. This means that application servers that are in the state of being recovered will not be able to accept any inbound work.

Important: WebSphere for z/OS uses the z/OS Resource Recovery Services (RRS) system function to provide the same transactional recovery functionality as is provided by the high availability peer recovery support on other platforms. Therefore, high availability peer recovery support is not available on a z/OS platform.

After the prerequisites are installed, starting a server on a system to which it was not configured implicitly places the server into peer restart and recovery mode. If you configured your XA Partner log to write to a non-shared HFS, or if you are using a JTA XA Resource Manager, you need to perform the following steps before starting a server:

1. (Required only if you are using a non-shared HFS.) Enable non-shared HFS support. When using a non-shared HFS, the configuration settings must be replicated across the different systems in the sysplex. This is done automatically by the deployment manager and node agent. To enable this support, each node agent in your configuration must be set as a recovery node. This change is made in the administrative console:
   a. In the administrative console navigation, select **System Administration** > **Node Agents**.
   b. Select a node agent from the list.
   c. Under **Additional Properties**, select **File Synchronization Service**.
   d. Under **Additional Properties**, select **Custom Properties**.
   e. Select **New**.
   f. Enter recoveryNode for **Name**, and true for **Value**. The **Description** field can be left blank.
   g. Repeat steps 3-7 for each node agent in your configuration.
   h. Save your configuration.
2. (Required only if you are using JTA XAResource Managers.) Make appropriate logs and classes are available on the alternate system If you plan to use WebSphere Application Server for z/OS peer restart and recovery, and your applications access JTA XAResource Managers, you must ensure that the appropriate logs and classes are available on the alternate system.

a. Point the WebSphere Application Server for z/OS variable TRANLOG_ROOT to a shared HFS. The TRANLOG_ROOT variable must point to a shared HFS, to which all systems in the WebSphere Application Server for z/OS cell can write. The XA partner log is stored here, and the alternate system must be able to read and update this log.

   Use the administrative console to set the WebSphere Application Server for z/OS variable, TRANLOG_ROOT, to the directory of a shared HFS, to which all systems in the WebSphere Application Server for z/OS cell can write.

   In the administrative console, click **Environment** > **Manage WebSphere Variables**. Then click on the **TRANLOG_ROOT** variable to bring up an new window in which you can specify the directory of the shared HFS.

b. Store the driver (i.e., JDBC Driver, JMS Provider, or JCA Resource Adapter, etc.) for each JTA XAResource Manager in an HFS that is readable by all systems in the WebSphere Application Server for z/OS cell. For example, if your connector is a JDBC driver for a database, the driver would likely be stored in a read-only HFS that is accessible by all systems in the sysplex. This allows the alternate system to read the saved classpath for the resource, and reconstruct it during a restart.

   If the connector used to access a JTA XAResource Manager is not stored in an HFS that is readable by all systems that might be used for recovery, when an application server restarts on an alternate system, it will either appear that there is no XA recovery work to do, or it will be impossible to load the classes necessary to communicate with the JTA XAResource Manager

3. Resolve InDoubt units.

   During a recovery, there will be instances when manual intervention is required to resolve InDoubt units. You will need to use RRS panels for this manual intervention.

Peer restart and recovery

The goal of every system is to have as little downtime as possible. Sometimes, however, system failures are inevitable. For example, a system failure might occur because the power unexpectedly goes out in your main system. When a system failure occurs, a restart action you can take is to restart on a peer system in the sysplex. This type of restart uses the *peer restart and recovery* function. Starting a server on a system to which it was not configured implicitly places it into peer restart and recovery mode.

Important: Peer Restart and Recovery (PRR) functionality is deprecated. You should use the integrated high availability support for the transaction service subcomponent, instead of Peer Restart and Recovery for transaction recovery. See Transactional high availability and Configuring transaction properties for peer recovery for more information about the integrated high availability support for the transaction service subcomponent and how to configure it for peer recovery of transactions being processed on a application server that fails.

When you experience a main system failure that results in InDoubt transactions with unknown outcomes, you need to obtain those intended transactional outcomes (ideally correctly) before the data can be utilized again. Peer restart and recovery provides an automated means of accomplishing this by restarting the controller on a peer system so that the ″locks″ that block the data can be dropped and the outcomes determined. This is in contrast to how a system usually handles a failure by automatically rolling back.

If a failure occurs, automatic restart management:

- Can restart WebSphere Application Server for z/OS and related servers on the same system, or
- Can use the WebSphere Application Server for z/OS peer restart and recovery function to restart related servers on an alternate system in the cell.

  WebSphere Application Server is not a recoverable *resource* manager. It is a recoverable *communication manager*. It has no recoverable locks of its own and it does not need to manage locks nor manage lock states in a log. It just needs to make sure that both callers and callees are connected in each of the communications sessions of a distributed transaction.

Peer restart and recovery restarts the controller on another system and goes through the transaction restart and recovery process so that we can assign outcomes to transactions that were in progress at the time of failure. During this transaction restart and recovery process, data might be temporarily inaccessible until the recovery process is complete. The restart and recovery process does not result in lost data.

Resource managers (such as DB2) that were being accessed at the time of failure may hold locks that are scoped to a transaction UR (unit of recovery). Once an outcome has been assigned to a UR, the resource managers will, generally, drop those locks.

When might PRR fail to recover servers

The major reason for recovery failure is if you experience a network outage while in the process of recovering. If the system cannot reach the superior or subordinate because the network is dead, communications cannot reestablish and the transaction cannot completely resolve.

Important: Peer Restart and Recovery (PRR) functionality is deprecated. You should use the integrated high availability support for the transaction service subcomponent, instead of Peer Restart and Recovery for transaction recovery. See Transactional high availability and Configuring transaction properties for peer recovery for more information about the integrated high availability support for the transaction service subcomponent and how to configure it for peer recovery of transactions being processed on a application server that fails.

When WebSphere Application Server for z/OS cannot automatically resolve all of the URs returned from RRS at restart, RRS will not allow the application server to move back to the home (original) system. If the application serve tries to go back while URs are still incomplete, you will receive an error code (C9C2186A) and a message describing an F02 return code from ATRIBRS. In order to get around this, manual resolution is required to mark the server for ″restart anywhere.″ RRS will do that once all of the URs in which WebSphere Application Server for z/OS is involved are *forgotten*. If RRS fails to mark the server *restart anywhere*, the server, upon failure, is required to start on the recovery system. This is not good because it doesn't allow you to move the server back to its true home system.

The ultimate goal of this is to resolve all transactions that the application server (the server instance-owned interests that could not complete recovery) is involved in, and then, if necessary, remove all of the application serve interests that remain in those URs. Once that is complete, browsing the RM data log will show if the resource manager is marked ″restart anywhere.″

You **want** to see:

```
RESOURCE MANAGER=BSS00.SY1.BBOASR4A.IBM
RESOURCE MANAGER MAY RESTART ON ANY SYSTEM
```

You do **not** want to see:

```
RESOURCE MANAGER=BSS00.SY2.BBOASR4A.IBM
RESOURCE MANAGER MUST RESTART ON SYSTEM SY2
```

Using RRS panels to resolve InDoubt units of recovery

Use this task to better understand messages received when using peer restart and recovery.

Important: Peer Restart and Recovery (PRR) functionality is deprecated. You should use the integrated high availability support for the transaction service subcomponent, instead of Peer Restart and Recovery for transaction recovery. See Transactional high availability and Configuring transaction properties for peer recovery for more information about the integrated high availability support for the transaction service subcomponent and how to configure it for peer recovery of transactions being processed on a application server that fails.

There are RRS version requirements that you must heed when using peer restart and recovery. For more information on these requirements, see *z/OS MVS Programming: Resource Recovery*.

If you receive the console message:

```
BBOT0015D OTS UNABLE TO RESOLVE ALL INCOMPLETE TRANSACTIONS FOR SERVER
string.  REPLY CONTINUE OR TERMINATE.
```

1. Note the server name specified for *string.* in the message.
2. Go to the SYSPRINT that is the status queue for that server, and search for messages BBOT0019 - BBOT0022 that refer to that server.
3. Read the resulting messages.
4. RRS does not allow an operator to resolve an InDoubt UR if the DSRM for that UR is active at the time. Therefore, you must stop the server. To do this, reply `TERMINATE` to the CONTINUE/TERMINATE WTOR.

The following non-console messages, which can be used to trigger automation, provide details about daemon activities when attempting restart and recovery:

- **BBOO003E** WEBSPHERE FOR z/OS CONTROL REGION string ENDED ABNORMALLY, REASON= *hstring*.
- **BBOO009E** WEBSPHERE FOR z/OS DAEMON string ENDED ABNORMALLY, REASON= *hstring*.
- **BBOO0171I** WEBSPHERE FOR z/OS CONTROL REGION string NOT STARTING ON CONFIGURED SYSTEM *string*
- The following messages, which are written only in recovery and restart mode, provide details about transactions that cannot be resolved during restart and recovery:
  - **BBOT0008I** TRANSACTION SERVICE RESTART INITIATED ON SERVER *string*
  - **BBOT0009I** TRANSACTION SERVICE RESTART UR STATUS COUNTS FOR SERVER string: IN-BACKOUT= *dstring*, IN-DOUBT= *dstring*, IN-COMMIT= *dstring*
  - **BBOT0010I** TRANSACTION SERVICE RESTART AND RECOVERY ON SERVER string IS COMPLETE
  - **BBOT0011I** SERVER *string* IS COLD STARTING WITH RRS
  - **BBOT0012I** SERVER *string* IS WARM STARTING WITH RRS
  - **BBOT0013I** TRANSACTION SERVICE RESTART AND RECOVERY ON SERVER *string* IS COMPLETE. THE SERVER IS STOPPING.
  - **BBOT0014I** TRANSACTION SERVICE RECOVERY PROCESSING FOR RRS URID ' *string* ' IN SERVER *string* IS COMPLETE.
  - **BBOT0016I** TRANSACTION SERVICE RESTART AND RECOVERY FOR SERVER *string* IS NOT COMPLETE. THE SERVER IS STOPPING DUE TO OPERATOR REPLY.
  - **BBOT0017I** TRANSACTION SERVICE RESTART AND RECOVERY FOR SERVER *string* IS CONTINUING DUE TO OPERATOR REPLY.
  - **BBOT0018I** TRANSACTION SERVICE RESTART AND RECOVERY FOR SERVER *string* IS STILL PROCESSING *dstring* INCOMPLETE UNIT(S) OF RECOVERY.
  - **BBOT0019W** UNABLE TO RESOLVE THE OUTCOME OF THE TRANSACTION BRANCH DESCRIBED BY URID: ' *string* ' XID FORMATID: ' *string* ' XID GTRID: ' *string* ' XID BQUAL: ' *string* ' BECAUSE THE OTS RECOVERY COORDINATOR FOR SERVER *string* ON HOST *string*: *dstring* COULD NOT BE REACHED.
  - **BBOT0020W** UNABLE TO PROVIDE THE SUBORDINATE OTS RESOURCE IN SERVER string ON HOST *string*: *dstring* WITH THE OUTCOME OF THE TRANSACTION DESCRIBED ON THIS SERVER BY URID: ' *string* ' XID FORMATID: ' *string*' XID GTRID: ' *string* ' XID BQUAL: ' *string* ' BECAUSE THIS SERVER HAS BEEN UNABLE TO RESOLVE THE OUTCOME WITH A SUPERIOR NODE.
  - **BBOT0021W** UNABLE TO *string* THE SUBORDINATE OTS RESOURCE IN SERVER *string* ON HOST string: dstring FOR THE TRANSACTION DESCRIBED ON THIS SERVER BY URID: ' *string* ' XID FORMATID: ' *string* ' XID GTRID: ' *string* ' XID BQUAL: ' *string* ' OR ANOTHER RESOURCE INVOLVED IN THIS UNIT OF RECOVERY BECAUSE ONE OR MORE RESOURCES COULD NOT BE REACHED OR HAVE NOT YET REPLIED.

- **BBOT0022W** UNABLE TO FORGET THE TRANSACTION WITH HEURISTIC OUTCOME DESCRIBED ON THIS SERVER BY URID: ' *string* ' XID FORMATID: '*string* ' XID GTRID: ' *string* ' XID BQUAL: ' *string* ' BECAUSE THE SUPERIOR COORDINATOR FOR SERVER *string* ON HOST *string*: *dstring* HAS NOT INVOKED FORGET ON THE REGISTERED RESOURCE.

Pay particular attention to the URID, XID FormatId, XID Gtrid, and XIDBqual attributes. You need to use these pieces of information when you manually resolve the relevant units of work via the RRS panels.

Resolving InDoubt units if you receive a BBOT00*xx*W message

If you receive a BBOT00*xx*W message, you must use RRS panels to view the outcome of other branches in the transaction and set the outcomes of InDoubt units to match the outcome of those other branches.

When a BBOT00*xx*W message is displayed, there is a possibility of an heuristic outcome. See *z/OS MVS Programming: Resource Recovery* for more information on how to use the RRS panels and what administrative access (RACF access to the facility class, for example) is needed to resolve URs and remove interests.

Perform the following steps to remove an expression of interest in this UR if you receive any of the following error messages:

- Messages BBOT0019W and BBOT0020W, which appear together, indicate that this server could not determine the outcome from its superior. Message BBOT0020W, describes the resource to which WebSphere Application Server for z/OS could not provide an outcome.
- Message BBOT0021W, which indicates that a server has determined the transaction outcome but has not been able to communicate it to its subordinates.
- Message BBOT0022W, which indicates that the transaction outcome was determined and communicated to the subordinate, but the subordinate resource has not been ″forgotten.″

1. Select option 3, ″Display/Update RRS Unit of Recovery information″ on the main RRS panel.
2. To view the details of this URID, enter it in the ″URID Pattern″ field on the query panel. Press **enter** to execute the query. The query results should display the UR. Take note of whether the state of this UR is InCommit, InBackout or InForget. In the column labeled S, enter `v` to display the details for this UR.
3. Remove the OTS interest. The RRS Unit of Recovery Details panel opens. Near the bottom of the panel, find the Expressions of Interest heading. This heading is followed by one or more rows that represent each individual expression of interest in this UR. Complete these steps to remove the OTS interest:
   a. Find the row that represents the OTS interest. This row will have an RM name in the form of `BSS00.`*`xxx`*`.`*`yyy`*`.IBM`, where *xxx* is the system to which the server was configured and *yyy* is the specific server name.
   b. Type `r` in the column labeled S to indicate that you want to remove this interest.
   c. Press the **Enter** key to execute the query.
4. Press the **Enter** key to confirm the removal of this interest. The RRS Remove Interest Confirmation panel opens. The RM name and UR identifier fields are pre-filled. Press the **Enter** key to confirm the removal of this interest.

You know you are done when RRS marks the subordinate server as restart anywhere. Determine this by choosing option 1 under Browse and RRS log stream, and then choosing sub-option 4 under RRS Resource Manager Data log.

Any subordinate nodes that restart and ask this server about this UR can not obtain this information. If you restart the server containing these nodes, they might be assigned an outcome that is different from the outcome of the transaction. You must manually resolve these nodes before you bring up the servers and start the server for which you just released the UR.

Resource Recovery Services Operations

This topic provides tips for using the z/OS Resource Recovery Services with WebSphere Application Server for z/OS.

Tips for RRS Operations

See *z/OS MVS Programming: Resource Recovery*, for RRS operations guidelines.

Tips for RRS operations:

- If you have configured your logstreams to the coupling facility, then monitor your log streams to ensure offload is not occurring. RRS will perform better if its recovery logs do not offload.

  Note: Proper sizing of the RRS logs is important. Too small and you get reduced throughput since logger is off-loading the logs too frequently. Too large and you could overflow your coupling facility.
- Keep the main and delayed (only contains active or live data) logs in your coupling facility. Make sure the CF definitions don't overflow.

  Note: A commit cannot occur until the log record is written.
- Until you stabilize your workloads, it is a good idea to use the archive log. If you have an archive log configured, RRS will unconditionally use it. However, there is a performance penalty for using it.

Recovering with JTA XAResource managers

When a JTA XAResource manager is enlisted in a global transaction, it cannot express an interest in the z/OS Resource Recovery Services unit of recovery (UR) like an RRS resource manager can. Instead, the WebSphere Application Server for z/OS transaction service will save information in its RRS interest indicating that a JTA Resource Manager was enlisted in the transaction.

Purpose

When you look at the UR through the RRS panels, you will not see an interest for each XA transaction branch, as you would for a resource manager like DB2 or CICS interest.

Because of the differences between RRS and JTA XAResource Managers, there is a different set of errors that can occur when dealing with a JTA XAResource. The following sections describe errors you might see when recovering with a JTA XAResource Manager. Some of these errors are expected, while others may indicate that there is another type of problem, such as connectivity, that needs to be addressed.

This topic describes Peer Restart and Recovery messages that are unique to the z/OS environment.

Messages

- **BBOT0025D:** OTS HAS ENCOUNTERED A LOG DATA MISMATCH. REPLY CONTINUE IF THIS IS EXPECTED OR TERMINATE IF UNEXPECTED.

  This message is issued when the restart epoch in the WebSphere Application Server for z/OS XA partner log does not match the restart epoch in RRS. These logs must remain in sync to guarantee the atomic outcomes of distributed transactions.

  If one or the other, but not both logs, were restored from a backup, a mismatch will occur. Since the XA partner log is maintained in the JVM, this error can also occur if the controller is started but then is canceled before the JVM is initialized. The RRS logstream will have been replayed before the XA partner log was initialized.

  This message gives the operator an opportunity to cancel recovery and determine why the logs are not in sync. If the machine is not in production and data integrity is not an issue, the operator may reply **CONTINUE** and recovery will attempt to complete with the mismatched logs. However, the results of this response are unpredictable. If the operator replies **TERMINATE**, the application server will shut down, and the problem can be investigated before completing recovery.

- **BBOT0026I:** TRANSACTION SERVICE RESTART AND RECOVERY FOR SERVER %s IS STILL PROCESSING AN UNKNOWN NUMBER OF XA TRANSACTIONS.

  This message is issued when the application server is unable to initiate contact with each JTA XAResource in its log. Since each JTA XAResource maintains its own logs, it is impossible to know how many transactions there are to recover. Look in the servant region for messages **WTRN0019**, and **WTRN0025**. These messages will help you determine what may be preventing the application server from communicating with these JTA XAResource Manager.

Setting up WebSphere Application Server for z/OS on multiple systems in a sysplex

A typical WebSphere Application Server for z/OS base runtime includes a cell with a location service daemon, called BBODMNB, and one node that includes an application server, called server1, that has a controller and any number of servants.

After you have installed the Application Server runtime and associated business application servers on a monoplex, you should evaluate whether you want to migrate the runtime and associated application servers to a sysplex configuration.

The benefits of migrating to a sysplex include:

- You can balance the workload across multiple systems, thus providing better performance management for your applications.
- As your workload grows, you can add new systems to meet demand, thus providing a scalable solution to your processing needs.
- By replicating the runtime and associated business application servers, you provide the necessary system redundancy to assure availability for your users. Thus, in the event of a failure on one system, you have other systems available for work.
- You can upgrade the Application Server from one release or service level to another without interrupting service to your users.

To enable WebSphere Application Server in a sysplex on the z/OS platform:

Table 5.

Subtask	Associated procedure (See . . .)
Setting up a sysplex	*z/OS MVS Setting Up a Sysplex*
Making decisions about the WebSphere Application Server for z/OS configuration and sysplexes	“Configuring WebSphere Application Server for a sysplex environment” on page 313
Preparing your security system	*Securing applications and their environment* PDF
Setting up data sharing	*DB2 Data Sharing: Planning and Administration*
Customizing base z/OS functions on the other systems in the sysplex	“Steps for customizing base z/OS functions on the other systems in the sysplex” on page 314
Making changes to TCP/IP	“Changing TCP/IP settings to support a sysplex environment” on page 316
	“Adding members to a cluster” on page 409

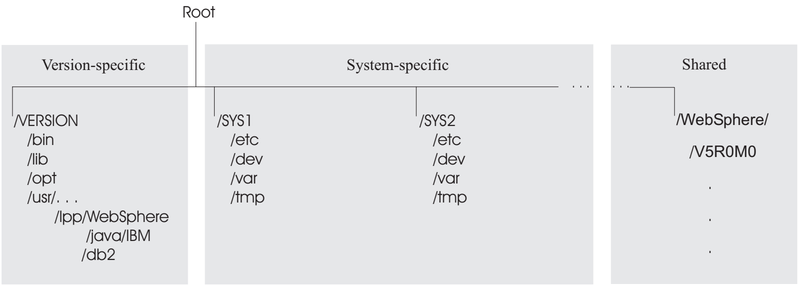
1. Set up a sysplex environment. The z/OS publication *z/OS MVS Setting Up a Sysplex* describes how to set up a z/OS sysplex.
2. Configure WebSphere Application Server for z/OS for a sysplex environment.
3. Prepare your security system.

4. Set up data sharing. Refer to the *DB2 Data Sharing: Planning and Administration* publication for the version of DB2 that is running on your z/OS system.
5. Customize base z/OS functions on the other systems in the sysplex.
6. Change the TCP/IP settings.
7. Define new application server clusters in the sysplex.

Overview of a WebSphere Application Server for z/OS sysplex

A typical WebSphere Application Server for z/OS base runtime includes a cell with a location service daemon, named BBODMNB, and a single node that includes an application server, named server1, a controller and one or more servants. After you have installed the WebSphere Application Server runtime and associated business application servers on a monoplex, you can migrate the runtime and associated application servers to a sysplex configuration.

The benefits of migrating to a sysplex include:
- You can balance the workload across multiple systems, thus providing better performance management for your applications.
- As your workload grows, you can add new systems to meet demand, thus providing a scalable solution to your processing needs.
- By replicating the runtime and associated business application servers, you provide the necessary system redundancy to assure availability for your users. Therefore, in the event of a failure on one system, you have other systems available for work.
- You can upgrade the Application Server from one release or service level to another without interrupting service to your users.



Configuring WebSphere Application Server for a sysplex environment

This task describes how to set up WebSphere Application Server for z/OS for a sysplex environment.

You should have completed the WebSphere Application Server for z/OS installation and customization on a monoplex or on a single system in a sysplex. Also, you must have enabled a z/OS sysplex. For more information on sysplexes, see the z/OS publication *z/OS MVS Setting Up a Sysplex*.

To configure WebSphere Application Server for z/OS for a sysplex environment:

1. Decide whether you want a single-system view of the error log. If you want a single-system view of the error log, and initially you set up the error log in the system logger and used DASD for logging, you must now configure the error log in the coupling facility.
2. Decide how you will share application executables in the cell.
3. Set up ARM. This release does not support cross-system restart, so you must set up your ARM policy accordingly. Make sure you specify TARGET_SYSTEM for the system on which each element runs (if you take the default TARGET_SYSTEM=*, you get cross-system restart).
4. Decide whether you will run all the WebSphere Application Server for z/OS run-time servers on every system in the cell.

   Recommendations: The following table provides recommendations and requirements for running servers in a cell.

Table 6. Running servers in a cell

Server	Recommendations and requirements for running servers in a cell
location service daemon and node agent	• You must run both the location service daemon and the node agent on each system in the cell in which you want WebSphere Application Server work to run. If some of the systems in your cell do not run WebSphere Application Server or WebSphere Application Server applications, you do not have to have the location service daemon and node agent on those systems. • If a server indicates that PassTickets are desirable for interaction with a client, you must run the location service daemon and node agent on the system where the z/OS client resides.
deployment manager	Make sure you follow the correct steps to configure a deployment manager cell.

5. **Optional:** Follow these steps to build WebSphere Application Server for z/OS deployment manager cells:
   a. Install the default application server on each node in your sysplex.
   b. Install a deployment manager cell on one node in your sysplex.
   c. Add default server nodes to the deployment manager cell.

Steps for customizing base z/OS functions on the other systems in the sysplex

Use these steps to change the base system.

You must have the WebSphere Application Server for z/OS product code installed through SMP/E, and you must have created copies of the product sample files.

Repeat the same customization to base z/OS functions that you did for your initial installation and customization of the Application Server. The steps are repeated here for convenience.

Note: The following steps assume that your default Application Server data set high-level qualifier (hlq) is `BBO`. If it is not, modify the examples to use your specified hlq.

Perform the following steps to change the base system:

1. Change SCHED*xx* to include the statements from the BBOSCHED sample file you ran in the customization dialog.
2. APF-authorize the BBO.SBBOLOAD, BBO.SBBOLD2, and BBO.SBBOLPA data sets.

   Example: Your PROG*xx* PARMLIB member could include:

```
APF FORMAT(DYNAMIC)
/*********************************************************************/
/* BOSS LOCAL DATASETS                                               */
/*********************************************************************/
APF ADD
    DSNAME(BBO.SBBOLOAD)
    VOLUME(vvvvvv)
APF ADD
    DSNAME(BBO.SBBOLD2)


    VOLUME(vvvvvv)
APF ADD
    DSNAME(BBO.SBBOLPA)
    VOLUME(vvvvvv)
```

   where *vvvvvv* is your volume identifier.

3. Ensure that the Language Environment (LE) data sets, SCEERUN and SCEERUN2, and the DB2 data set, SDSNLOAD, are authorized.
4. Do **not** APF-authorize BBO.SBBOULIB or BBO.SBBOMIG, because they should run under the authority of the client user.
5. Place the Application Server modules. Use the following table to place Application Server modules:

Table 7. Placing modules in LPA or link list

Modules	Notes
BBO.SBBOLPA	Load all members into the LPA.
BBO.SBBOLOAD	We recommend you dynamically load all members into the LPA. If your virtual storage is constrained, place the members in the link list.
BBO.SBBOMIG	You can put members into the link list or LPA.
BBO.SBBOLD2	Do **not** put members from SBBOLD2 in the LPA. Place these members in the link list.
BBO.SBBOULIB	Do **not** place these members in **either** the LPA or link list.

Rule: These data sets are PDSEs and cannot be added to members in LPALST*xx* or IEALPA*xx*.

Recommendation: For automation, if you want to ensure the Application Server modules are loaded into dynamic LPA and available after an IPL, create a new PROG*xx* member with the SETPROG LPA commands and invoke the PROG*xx* member from PARMLIB COMMND*xx*.

Example:

```
SETPROG LPA,ADD,MASK=*,DSNAME=
BBO.
SBBOLOAD
SETPROG LPA,ADD,MASK=*,DSNAME=
BBO.
SBBOLPA
```

- Change BBO to the appropriate qualifier if BBO is not the high-level qualifier for your Application Server data sets.
- If you are using SETPROG on a running system, before adding modules BBO.SBBOLPA, BBO.SBBOLOAD, or BBO.SBBOMIG, be sure to purge any already existing modules with the same names from the LPA.

6. **Optional:** If you used a PROG*xx* file for APF authorizations or the LPA, be sure to issue:

   ```
   SET PROG=xx
   ```

7. **Optional:** Make sure all the BBO.* data sets are cataloged. While not required, this is highly recommended.
8. Update your SYS1.PARMLIB(BLSCUSER) member with the IPCS models supplied by member BBOIPCSP. For details in BLSCUSER, see *z/OS MVS IPCS User's Guide*.
9. **Optional:** Start SMF recording. If you want to start SMF recording to collect system and job-related information on the WebSphere Application Server for z/OS system:
   a. Edit the SMFPRM*xx* parmlib member.
      1) Insert an 'ACTIVE' statement to indicate SMF recording.
      2) Insert a SYS statement to indicate the types of SMF records you want the system to create.

         Example: Use SYS(TYPE(120:120)) to select type 120 records only. Keep the number of selected record types small, to minimize the performance impact.
   b. To start writing records to DASD, issue the following command:

      ```
      t smf=xx
      ```

      Where xx is the suffix of the SMF parmlib member (SMFPRM*xx*). For more information about the SMF parmlib member, see *z/OS MVS System Management Facilities (SMF)*.

      When you activate writing to DASD, the data is recorded in a data set (specified in SMFPRM*xx*).

Changing TCP/IP settings to support a sysplex environment

You must have TCP/IP installed and configured.

You might need to make the following TCP/IP changes if you are running WebSphere Application Server for z/OS in a sysplex environment:

1. Change DNS entries. Assuming you use an implementation of the DNS that allows use of generic IP names that dynamically resolve to like-configured servers, you must adjust the IP names in your DNS. Keep the generic IP name of the location service daemon, but add a new IP name for the second and subsequent location service daemon servers. This is important not only for workload balancing, but in the event of a server failure: the DNS can direct work to other servers.
2. In the TCP/IP profile for each additional system in the cell, add a port for the location service daemon, and associate that port with a new location service daemon server name.

   By default, WebSphere Application Server for z/OS uses port 5655 for the location service daemon. WebSphere Application Server for z/OS also names the first location service daemon server DAEMON01 and increments the suffix on that name for each new location service daemon server; for example, DAEMON02, DAEMON03, and so forth. Therefore, on your second system in the cell, add a port and associate it with DAEMON02.

   Example:

   ```
   5655
      TCP     DAEMON02
   ```

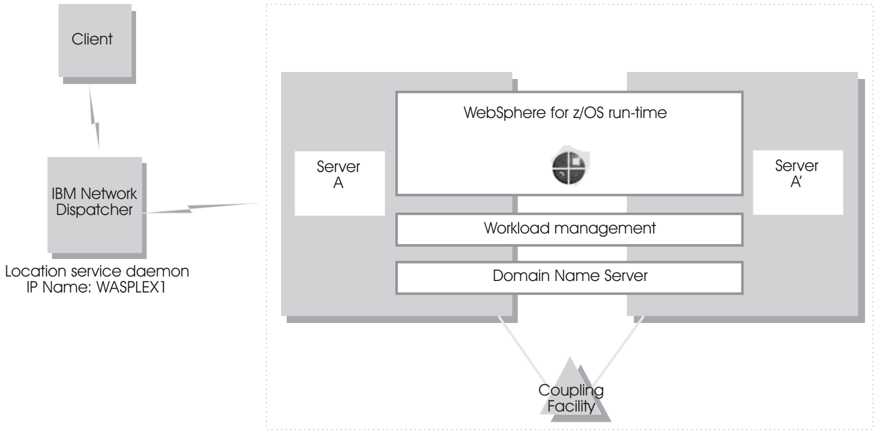
   Follow the same pattern for the third and subsequent systems in the cell.

Load Balancer

Load Balancer (known as Network Dispatcher in earlier releases) is a router that handles network requests for the cell.

Characteristics of such a configuration are:

- The location service daemon IP Name is associated with the IP address of the router.
- Load Balancer cooperates with workload management to route requests through the cell. The client never sees a change in IP addresses.
- The implication for clients is that they can cache the IP addresses, because this configuration does not change them dynamically.



Configuring transport chains

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 or HTTP. Network ports can be shared among all of the channels within a chain. The channel framework function automatically distributes a request arriving on that port to the correct I/O protocol channel for processing.

Ensure that a port is available for the new transport chain. If you need to set up a shared port, you must:

- Use wsadmin commands to create your transport chain.
- Make sure that all channels sharing that port have the same discrimination weight assigned to them.

You need to configure transport chains to provide networking services to such functions as the service integration bus component of IBM service integration technologies, WebSphere Secure Caching Proxy, and the high availability manager core group bridge service.

You can either use the administrative console or wsadmin commands to create a transport chain. If you want to use wsadmin commands, see the *Using the administrative clients* PDF for more information. If you use the administrative console, complete the following steps:

1. In the administrative console, click **Servers > Application servers >** *server_name*, and then select one of the following options, depending on the type of chain you are creating:
   - Under **SIP Container Settings**, click **SIP container transport chains**.
   - Under **Web container settings**, click **Web container transport chains**.
   - Under **Container services**, click **ORB Service > ORB Service Transport Chains**.
   - Under **Server messaging**, click either **Messaging engine inbound transports** or **WebSphere MQ link inbound transports**.
2. Click **New**. The Create New Transport Chain wizard initializes. During the transport chain creation process, you are asked to:
   - Specify a name for the new chain.
   - Select a transport chain template
   - Select a port, if one is available to which the new transport chain is bound. If a port is not available or you want to define a new port, specify a port name, the host name or IP address for that port, and a valid port number.

   When you click **Finish**, the new transport chain is added to the list of defined transport chains on the **Transport chain** panel.
3. Click the name of a transport chain to view the configuration settings that are in effect for the transport channels contained in that chain. To change any of these settings:
   a. Click the name of the channel whose settings you need to change.
   b. Change the configuration settings. Some of the settings, such as the port number are determined by what is specified for the transport chain when it is created and cannot be changed.
   c. Click on **Custom properties** to set any custom properties that are defined for your system.
4. When you your configuration changes, click **OK**.
5. Stop the application server and start it again.

   You must stop the application server and start it again before your changes take effect.

Update any routines you have that issue a call to start transports during server startup. When a routine issues a call to start transports during server startup, WebSphere Application Server issues an error message.

Transport chains

Transport chains represent a network protocol stack that is used for I/O operations within an application server environment. Transport chains are part of the channel framework function that provides a common networking service for all components, including the service integration bus component of IBM service integration technologies, WebSphere Secure Caching Proxy, and the high availability manager core group bridge service.

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, DCS. or HTTP. Network ports can be shared among all of the channels within a chain. The channel framework function automatically distributes a request arriving on that port to the correct I/O protocol channel for processing.

Important: Transport chains are not as streamlined as the native HTTP transports. Therefore, you should continue to use HTTP transports instead of transport chains unless you want to take advantage of IPv6 or Web Services Atomic Transaction (WS-AT) support , or if you have multiple ports configured for the Application Server.

transition: If you have a routine that issues a call to start transports during server startup, unless you have a mixed-node environment and that server is running in either a V5.1 or V6.0.x, you must modify your routine to issue a call to start transport chains instead of the transports. WebSphere Application Server issues an error message if it receives a call to start transports for a server that is not running in a V5.x or V6.0.x node.

The transport chain configuration settings determine which I/O protocols are supported for that chain. Following are some of the more common types of channels. Custom channels that support requirements unique to a particular customer or environment can also be added to a transport chain.

DCS channel
Used by the core group bridge service, the data replication service (DRS), and the high availability manager to transfer data, objects, or events among application servers.

HTTP inbound channel
Used to enable communication with remote servers. It implements the HTTP 1.0 and 1.1 standards and is used by other channels, such as the Web container channel, to serve HTTP requests and to send HTTP specific information to servlets expecting this type of information.

HTTP inbound channels are used instead of HTTP transports to establish the request queue between a WebSphere Application Server plug-in for Web servers and a Web container in which the Web modules of an application reside.

HTTP proxy inbound channel
Used to handle HTTP requests between a proxy server and application server nodes.

HTTP Tunnel channel
Used to provide client applications with persistent HTTP connections to remote hosts that are either blocked by firewalls or require an HTTP proxy server (including authentication) or both. An HTTP Tunnel channel enables the exchange of application data in the body of an HTTP request or response that is sent to or received from a remote server. An HTTP Tunnel channel also enables client-side applications to poll the remote host and to use HTTP requests to either send data from the client or to receive data from an application server. In either case, neither the client nor the application server is aware that HTTP is being used to exchange the data.

JFAP channel
Used by the Java Message Service (JMS) server to create connections to JMS resources on a service integration bus.

MQ channel
Used in combination with other channels, such as a TCP channel, within the confines of WebSphere MQ support to facilitate communications between a WebSphere System Integration Bus and a WebSphere MQ client or queue manager.

ORB Service channel
Used in combination with other channels, such as a TCP channel, to handle CORBA and

RMI/IIOP messages for the ORB Service. It enables clients to make requests and receive responses from servers in a network-distributed environment.

SIP channel
Used to create a bridge in the transport chain between a session initiation protocol (SIP) inbound channel, and a servlet and JavaServer Page engine.

SIP container inbound channel
Used to handle communication between the SIP inbound channel and the SIP servlet container.

SIP inbound channel
Used to handle inbound SIP requests from a remote client.

SSL channel
Used to associate an Secure Sockets Layer (SSL) configuration repertoire with the transport chain. This channel is only available when SSL support is enabled for the transport chain. An SSL configuration repertoire is defined in the administrative console, under security, on the **SSL configuration repertoires > SSL configuration repertoires** page.

TCP channel
Used to provide client applications with persistent connections within a Local Area Network (LAN) when a node uses transmission control protocol (TCP) to retrieve information from a network.

UDP channel
Used to provide client applications with persistent connections within a Local Area Network (LAN) when a node uses user datagram protocol (UDP) to retrieve information from a network.

Web container channel
Used to create a bridge in the transport chain between an HTTP inbound channel and a servlet and JavaServer Page (JSP) engine.

HTTP transport collection

Use this page to view or manage HTTP transports. Transports provide request queues between WebSphere Application Server plug-ins for Web servers and Web containers in which the Web modules of applications reside. When you request an application in a Web browser, the request is passed to the Web server, then along the transport to the Web container.

Important: You can use HTTP transports only on a Version 5.1 node in a mixed WebSphere Application Server environment. You must use HTTP transport channels instead of HTTP transports to handle your HTTP requests on all of your other nodes.

The use of IPv6 (Internet Protocol Version 6) and WS-AT (Web Services Atomic Transactions) are not supported on HTTP transports; they are only supported on HTTP transport channel chains.

To view the HTTP Transport administrative console page, click **Servers > Application Servers >** *server_name* **> Web Container Settings > Web Container > HTTP Transports**.

Host

Specifies the host IP address to bind for transport. If the application server is on a local machine, the host name might be `localhost`.

Port

Specifies the port to bind for transport. The port number can be any port that currently is not in use on the system. The port number must be unique for each application server instance on a given machine.

For the z/OS platform, a maximum of two ports, one for HTTP requests and one for HTTPS requests, is allowed for each process configured as an HTTP transport. Additional ports can be configured as HTTP transport chains Additional ports can be configured as HTTP transport channels.

SSL Enabled

Specifies whether to protect connections between the WebSphere plug-in and application server with Secure Sockets Layer (SSL). The default is not to use SSL.

HTTP transport settings

Use this page to view and configure an HTTP transport. The name of the page might be that of an SSL setting such as DefaultSSLSettings. This page will not be visible if you do not have an HTTP transport defined for your system.

Important: You can use HTTP transports only on a V5.1 node in a mixed WebSphere Application Server environment. You must use HTTP transport channels instead of HTTP transports to handle your HTTP requests on all of your other nodes.

The use of IPv6 (Internet Protocol Version 6) and WS-AT (Web Services Atomic Transactions) are not supported on HTTP transports; they are only supported on HTTP transport channel chains.

To view the HTTP Transport panel on the administrative console, click **Servers** > **Application Servers** > *server_name* > **Web Container Settings** > **Web Container** > **HTTP Transports** > *host_name*.

Host

Specifies the host IP address to bind for transport.

If the application server is on a local machine, the host name might be `localhost`.

Data type	String

Port

Specifies the port to bind for transport. Specify a port number between 1 and 65535. The port number must be unique for each application server on a given machine.

Data type	Integer
Range	1 to 65535

SSL Enabled

Specifies whether to protect connections between the WebSphere Application Server plug-in and application server with Secure Sockets Layer (SSL). The default is not to use SSL.

Data type	Boolean
Default	false

SSL

Specifies the Secure Sockets Layer (SSL) settings type for connections between the WebSphere Application Server plug-in and application server. The options include one or more SSL settings defined in the Security Center; for example, DefaultSSLSettings, ORBSSLSettings, or LDAPSSLSettings.

Data type	String
Default	An SSL setting defined in the Security Center

HTTP transport custom properties

Use this page to set custom properties for an HTTP transport.

Important: You can use HTTP transports only on a V5.1 node in a mixed WebSphere Application Server environment. You must use HTTP transport channels instead of HTTP transports to handle your HTTP requests on all of your other nodes.

The use of IPv6 (Internet Protocol Version 6) and WS-AT (Web Services Atomic Transactions) are not supported on HTTP transports; they are only supported on HTTP transport channel chains.

If you are using HTTP transports, you can set the following custom properties on either the Web Container or HTTP Transport Custom Properties page on the administrative console. When set on the Web container Custom Properties page, all transports inherit the properties. Setting the same properties on a transport overrides like settings defined for a Web container.

To specify custom properties for a specific transport on the HTTP Transport:

1. In the administrative console click **Servers** > **Application Servers** > *server_name* > **Web Container settings** > **Web Container** >**HTTP Transport**
2. Select a host.
3. Under **Additional Properties** select **Custom Properties**.
4. On the Custom Properties page, click **New**.
5. On the settings page, enter the property you want to configure in the **Name** field and the value you want to set it to in the **Value** field.
6. Click **Apply** or **OK**.
7. Click **Save** on the console task bar to save your configuration changes.
8. Restart the server.

Following is a list of custom properties provided with the Application Server. These properties are not shown on the settings page for an HTTP transport.

ConnectionIOTimeOut:

Use the `ConnectionIOTimeOut` property to specify how long the J2EE server waits for an I/O operation to complete. Set this variable for each of the HTTP transport definitions on the server. You will need to set this variable for both SSL transport and non-SSL transport. Specifying a value of zero disables the time out function.

Data type	Integer
Default	For the z/OS platform: 120 seconds

ConnectionKeepAliveTimeout:

Use the `ConnectionKeepAliveTimeout` property to specify the maximum number of seconds to wait for the next request on a keep alive connection.

Data type	Integer
Default	For the z/OS platform: 30 seconds

ConnectionResponseTimeout:

This property is only valid in a z/OS environment. Use the ConnectionResponseTimeout property to set the maximum amount of time, in seconds, that the J2EE server will wait for an application component to respond to an HTTP request. Set this variable for each of the HTTP transport definitions on the server. You will need to set this variable for both SSL transport and non-SSL transport. If the response is not received within the specified length of time, the servant (region) might fail with ABEND EC3 and

RSN=04130007. Setting this timer prevents client applications from waiting for a response from an application component that might be deadlocked, looping, or encountering other processing problems that cause the application component to hang.

Use the server custom properties `protocol_http_timeout_output_recovery` and `protocol_https_timeout_output_recovery`, described in this article, to indicate the recovery action that you want taken on timeouts for requests received over the HTTP and HTTPS transports.

Data type	Integer
Default	300 seconds

MaxKeepAliveRequests:

Use the `MaxKeepAliveRequests` property to specify the maximum number of requests which can be processed on a single keep alive connection. This parameter can help prevent denial of service attacks when a client tries to hold on to a keep-alive connection. The Web server plug-in keeps connections open to the application server as long as it can, providing optimum performance.

On the z/OS platform, when this property is set to 0 (zero), the connection is closed after every request.

Data type	Integer
Default	50 requests

MutualAuthCBindCheck:

This property is only valid on the z/OS platform. Use the `MutualAuthCBindCheck` property to specify whether or not a client certificate should be resolved to a SAF principal. If this property is set to `true`, all SSL connections from a browser must have a client certificate, and the user ID associated with that client certificate must have RACF CONTROL authority for CB.BIND.servername. If these conditions are not met, the connection will be closed. Issue the following RACF command to give the user ID associated with that client certificate RACF CONTROL authority:

```
PERMIT CB.BIND.servername CLASS(CBIND) ID(clientCertUserid) ACCESS(CONTROL)
```

Data type	String
Value	true or false
Default	false

RemoveServerHeader: Use this property to specify whether an existing server header is removed before a response message is sent. If this property is set to `true`, the value specified for the ServerHeaderValue property is ignored.

Data type	String
Value	true or false
Default	false

ResponseBufferSize:

This property is only valid for the z/OS platform. It is used to specify, in bytes, the default size of the initial buffer allocation for the response buffer. When the buffer fills up, a flush for this buffer space will automatically occur. If a value is not specified for this property, the default response buffer size of 32K bytes is used.

The setBufferSize() API method can be used to override the value specified for this custom property at the individual servlet level.

Data type	Integer
Default	32000 bytes

ServerHeader:

This property is only valid for the z/OS platform. Use the `ServerHeader` property to suppress the server HTTP header (Server:) in responses. When the server header custom property is not specified, the default is equal to a setting of **true** and the server header is included in the HTTP response. Set this property to **false** if you want to prevent the inclusion of the server header.

Data type	String
Value	true or false
Default	true

ServerHeaderValue: Use this property to specify a server header this is added to outgoing response messages if server header is not already provided. This property is ignored if the RemoveServerHeader property is set to true.

Data type	string
Default	WebSphere Application Server/*x.x* *x.x* is the version of WebSphere Application Server that you are using.

SoLingerValue: Use this property to specify, in seconds, the amount, that the socket close operation waits for data contained in the TCP/IP send buffer to be sent. This property is ignored if the UseSoLinger property is set to false.

Data type	Integer
Default	20 seconds

TcpNoDelay: Use this property to set the socket TCP_NODELAY option which enables and disables the use of the TCP Nagle algorithm for connections received on this transport. When this property is set to `true`, use of the Nagle algorithm is disabled.

Data type	String
Value	true or false
Default	true

TrustedProxy:

This property is only valid for the z/OS platform. Use the `TrustedProxy` property to indicate that the application server can use the private headers that the Web server plug-in adds to requests.

Data type	String
Default	false

UseSoLinger: Use this property to set the socket SO_LINGER option. This property configures whether the socket close operation waits until all of the data contained in the TCP/IP send buffer is sent before closing a connection. If this property is enabled, and the time expires before the all of the content of the send buffer sent, any data remaining in the send buffer is lost.

The SoLingerValue property is ignored if this property is set to false.

Data type	String
Value	true or false
Default	true

HTTP transport channel custom properties

If you are using an HTTP transport channel, you can add the following custom property to the configuration settings for that HTTP transport channel.

To add a custom property:

1. In the administrative console, click **Application servers >** *server_name* **Web container settings > Web container transport chains >** *chain_name* **> HTTP Inbound Channel > 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. Restart the server.

Following is a list of custom properties provided with the application server. These properties are not shown on the settings page for an HTTP transport channel.

inProcessLogFilenamePrefix

Use the inProcessLogFilenamePrefix property to specify a prefix for the filename of the network log file. Normally, when inprocess optimization is enabled, requests through the inprocess path are logged based on the logging attributes set up for the Web container's network channel chain. You can use this property to add a prefix to the filename of the network log file. This new filename is then used as the filename for the log file for inprocess requests. Requests sent through the inprocess path are logged to this file instead of to the network log file. For example, if the log file for a network transport chain is named .../httpaccess.log, and this property is set to local for the HTTP channel in that chain, the filename of the log file for inprocess requests to the host associated with that chain is .../localhttpaccess.log.

Data type	String

listenBacklog

Use the listenBacklog property to specify the maximum number of outstanding connect requests that the operating system will buffer while it waits for the application server to accept the connections. If a client attempts to connect when this operating system buffer is full, the connect request will be rejected. Set this value to the number of concurrent connections that you would like to allow. Keep in mind that a single client browser might need to open multiple concurrent connection; also keep in mind that increasing this value consumes more kernel resources. The value of this property is specific to each transport.

Data type	Integer
Default	511

HTTP Tunnel transport channel custom property

If you are using an HTTP Tunnel transport channel, you can add the following custom property to the configuration settings for that HTTP Tunnel transport channel.

To add a custom property:

1. In the administrative console, click **Servers > Application servers >** *server_name* **> Ports**. Click on **View associated transports** for the HTTP Tunnel port to whose configuration settings you want to add this custom property.
2. Click **New**.
3. 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.
4. Click **Apply** or **OK**.
5. Click **Save** to save your configuration changes.
6. Restart the server.

Following is a description of the custom property that is provided with the application server. This property is not shown on the settings page for an HTTP Tunnel transport channel.

pluginConfigurable

Indicates whether or not the configuration settings for the HTTP Tunnel transport channel are included in the plugin-cfg.xml file for the Web server associated with the application server that is using this channel. Configuration settings for each of the Web container transport channels defined for an application server are automatically included in the plugin-cfg.xml file for the Web server associated with that application server.

Data type	Boolean
Default	False

Web container transport custom properties

Use this page to set custom properties for a Web container transport.

Unless you are not migrating from an previous version of WebSphere Application Server, you must use HTTP transport channels instead of HTTP transports to handle your HTTP requests.

If you are using Web container transports, you can set the following custom properties on the Web Container **Custom Properties** panel on the administrative console. When set on the Web container **Custom Properties** page, all transports inherit the properties. Setting the same properties on a transport overrides like settings defined for a Web container.

To specify custom properties for a specific transport on the HTTP Transport:

1. In the administrative console click **Servers** > **Application Servers** > *server_name* > **Web Container settings** > **Web Container**
2. Select a host.
3. Under **Additional Properties** select **Custom Properties**.
4. On the Custom Properties page, click **New**.
5. On the settings page, enter the property you want to configure in the **Name** field and the value you want to set it to in the **Value** field.
6. Click **Apply** or **OK**.
7. Click **Save** on the console task bar to save your configuration changes.
8. Restart the server.

Following is a list of custom properties provided with the Application Server. These properties are not shown on the settings page for a Web container transport.

disableRequestMessageChunking

This custom property disables request message chunking when set to true. All the request body up to `maxRequestMessageBodySize` is buffered in memory.

Value
True or False. When a value is not specified, the default value is false.

maxRequestMessageBodySize

If `disableRequestMessageChunking` is false, this is the maximum amount of request body that is buffered in memory before sending the next chunk to the SR. If `disableRequestMessageChunking` is true, this is the maximum amount of request body data that is buffered in memory before sending the complete request to the SR. An HTTP 404 is returned if `maxRequestMessageBodySize` is exceeded.

Value
If `disableRequestMessageChunking` is false, the default size is 32K and maximum size is 10MB. If `disableRequestMessageChunking` is true, the default size is 10MB and the maximum size is 100MB.

Configuring inbound HTTP request chunking

Inbound HTTP request chunking, is configured at the Web container transport chain level. You can configure each Web container chain to enable or disable chunking. You can also configure the maximum message size for chunking disabled and the maximum chunk size for chunking enabled for each chain.

See the page for details on these settings.

The chains that host the Integrated Solutions console have chunking enabled by default, while all other Web container chains have chunking disabled by default. The reason for this is that there are some limitations in the use of inbound HTTP chunking on WebSphere Application Server for z/OS V6.1.

The following are limitations of inbound HTTP chunking

- The applications that are served by chains with chunking enabled must support chunked HTTP encoding. See RFC 2616 for more information on chunked encoding. If your application does not support chunked encoding, then you must map it to a Web container chain with chunking disabled.
- The current implementation requires a Web application to read the entire request, both headers and body, before sending any response data back to the client. If the Web application does not read the entire request, this results in an error in the servlet as well as an HTTP 500 Internal Server Error returned to the client.

1. In the administrative console click **Servers > Application Servers >** *server_name* **> Web Container settings > Web Container**
2. Select a host.
3. **Optional:** Enable request message chunking. See the article, "Web container transport custom properties" on page 325 for details on these settings.
   a. Under **Additional Properties** select **Custom Properties**.
   b. On the Custom Properties page, click **New**.
   c. On the settings page, enter the property, **disableRequestMessageChunking** in the **Name** field and the enter the value false in the **Value** field.
   d. Specify the maximum amount of request body that is buffered in memory before sending the next chunk to the SR.
   e. Click **Apply** or **OK**.
4. **Optional:** Disable request message chunking. See the article, "Web container transport custom properties" on page 325 for details on these settings.
   a. Under **Additional Properties** select **Custom Properties**.
   b. On the Custom Properties page, click **New**.

c. On the settings page, enter the property, **disableRequestMessageChunking** in the **Name** field and the enter the value true in the **Value** field.

d. Specify the maximum amount of request body data that is buffered in memory before sending the complete request to the SR.

e. Click **Apply** or **OK**.

5. Click **Save** on the console task bar to save your configuration changes.
6. Restart the server.

Transport chain problems

Review the following topics if you encounter a transport chain problem.

TCP transport channel fails to bind to a specific host/port combination

If a TCP transport channel fails to bind to a specific port, one of the following situations might have occurred:

- You are trying to bind the channel to a port that is already bound to another application, such as another instance of a WebSphere Application Server.
- You are trying to bind to a port that is in a transitional state waiting for closure. This socket must transition to closed before you restart the server. The port might be in TIME_WAIT, FIN_WAIT_2, or CLOSE_WAIT state. Issue the `netstat -a` command from a command prompt to display the state of the port to which you are trying to bind.

Deleting a transport chain

Transport chains cannot be deleted the same way that HTTP transports can be deleted. Because you cannot have multiple HTTP transports associated with the same port, when you delete an HTTP transport, you effectively delete the associated port and stop all traffic on that port. However, the process is more complicated for a transport chain because multiple transport chains might be associated with the same port and you do not want to disrupt traffic on transport chains that you are not deleting.

Determine whether you want to delete a particular transport chain or all of the transport chains that are associated with a specific port.

You might have to delete one or more transport chains if you have to delete a port.

To delete a transport chain:

1. In the administrative console, click **Servers > Application servers >** *server* **> Ports**.
2. In the list of available ports, locate the port that you want to delete and click **View associated transports** for that port.
3. Select the transport chain you want to delete, and click **Delete**. If you intend to delete the port that is associated with this transport chain, repeat this step for all of the transport chains associated with this port.
4. Click **Save** to save your changes.

If you delete all of the transport chains associated with a port, you can delete the port.

Disabling ports and their associated transport chains

Transport chains cannot be disabled the same way that HTTP transports can be disabled. Because you cannot have multiple HTTP transports associated with the same port, when you disable an HTTP transport, you effectively disable the associated port and stop all traffic on that port. However, the process is more complicated for a port that has associated transport chains because multiple transport chains might be associated with the same port, and you might not want to disrupt traffic on all of the transport chains at the same time.

Determine whether you want to disable a particular transport chain or all of the transport chains that are associated with a specific port.

You might need to disable a transport chain if you want to temporarily stop all incoming traffic on a particular port or on a particular transport chain that is associated with that port.

To disable a specific transport chain:

1. In the administrative console, click **Servers** > **Application servers** > *server* > **Ports**.
2. In the list of available ports, locate the port that you want to delete and click **View associated transports** for that port.
3. Click the transport chain you want to disable.
4. Unselect the **Enabled** field, and click **OK**. If you want to temporarily stop all of the incoming traffic on a port, repeat this step for all of the transport chains associated with this port.
5. Click **Save** to save your changes.

When you want traffic to resume on these disabled transport chains, repeat the preceding steps for all of the transport chains you disabled, and select the **Enabled** field.

Transport chains collection

Use this page to view or manage transport chains. Transport chains enable communication through transports, or protocol stacks, which are usually socket based.

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 or HTTP. Network ports can be shared among all of the channels within a chain. The Channel Framework function automatically distributes a request arriving on that port to the correct I/O protocol channel for processing.

Important: On the z/OS platform, transport chains are not as streamlined as the native HTTP transports. Therefore, you should continue to use HTTP transports instead of transport chains unless you want to take advantage of IPv6 or Web Services Atomic Transaction (WS-AT) support , or if you want multiple HTTP and HTTPS ports configured for the Application Server.

The **Transport chains** page lists the transport chains defined for the selected application server. Transport chains represent network protocol stacks operating within this application server.

To view this administrative console page, click **Servers** > **Application servers** > *server_name* > **Ports**. Click on **View associated transports** for the port whose transport chains you want to view.

Name

Specifies a unique identifier for the transport chain. The name must consist of alphanumeric or national language characters and can start with a number. The name must be unique within a WebSphere Application Server configuration. Click on the name of a transport chain to change its configuration settings.

Enabled

When set to true, indicates that the transport chain is activated at application server startup.

Host

Specifies the host IP address to bind for transport. If the application server is on a local machine, the host name might be localhost.

Port

Specifies the port to bind for transport. The port number can be any port that currently is not in use on the system, might be localhost or the wildcard character * (an asterisk). The port number must be unique for each application server instance on a given machine

SSL Enabled

When enabled, users are notified that there is a channel that enables Secure Sockets Layer (SSL) in the listed chain. When SSL is enabled, all traffic going through this transport is encrypted and digitally secured.

Transport chain settings

Use this page to view a list of the types of transport channels configured for the selected 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.

To view this administrative console page, click **Servers** > **Application servers** > *server_name* > **Ports**. Click on **View associated transports** for the port whose transport chains you want view and then click on the name of a specific chain.

Name

Specifies the name of the selected transport chain.

You can edit this field to rename this transport chain. However, remember that the name must be unique within a WebSphere Application Server configuration.

Enabled

When checked, this transport chain is activated at application server startup.

Transport channels

Lists the transport channels configured for this transport chain and their configuration settings. To change a transport channel's configuration settings, click on the name of that transport channel.

HTTP tunnel transport channel settings

Use this page to view and configure an HTTP tunnel transport channels. Inbound connections sent through this channel are tunneled over HTTP, allowing intermediates to view this data as the body of an HTTP message instead of in its natural format. This type of channel is often used to circumvent firewalls with protocol restrictions.

To view this administrative console page, click **Servers** > **Application servers** > *server_name* > **Ports** . Click on **View associated transports** for the port associated with the HTTP Tunnel transport channel whose settings you want to look at.

Transport channel name

Specifies the name of the HTTP tunnel transport 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 tunnel transport 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

HTTP transport channel settings

Use this page to view and configure an HTTP transport channel. This type of transport channel handles HTTP requests from a remote client.

An HTTP transport channel parses HTTP requests and then finds an appropriate application channel to handle the request and send a response.

To view this administrative console page, click **Servers > Application servers >** *server_name* **> Ports** . Click on **View associated transports** for the port associated with the HTTP transport channel whose settings you want to look at.

Transport channel name

Specifies the name of the HTTP transport 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 transport 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

Maximum persistent requests

Specifies the maximum number of persistent (keep-alive) requests that are allowed on a single HTTP connection. If a value of 0 (zero) is specified, only one request is allowed per connection. If a value of -1 is specified, an unlimited number of requests is allowed per connection.

Data type	Integer
Default	100

Use persistent (keep-alive) connections

When selected, the HTTP transport channel, when sending an outgoing HTTP message, uses a persistent (keep-alive) connection instead of a connection that closes after one request or response exchange occurs.

Note: If a value other than 0 is specified for the maximum persistent requests property, the Use persistent (keep-alive) connections property setting is ignored.

The default for this property is selected.

Read timeout

Specifies the amount of time, in seconds, the HTTP transport channel waits for a read request to complete on a socket after the first read request occurs. The read being waited for could be an HTTP body (such as a POST) or part of the headers if they were not all read as part of the first read request on the socket.

Data type	Integer
Default	60 seconds

Write timeout

Specifies the amount of time, in seconds, that the HTTP transport channel waits on a socket for each portion of response data to be transmitted. This timeout usually only occurs in situations where the writes are lagging behind new requests. This can occur when a client has a low data rate or the server's network interface card (NIC) is saturated with I/O.

Data type	Integer
Default	60 seconds

Persistent timeout

Specifies the amount of time, in seconds, that the HTTP transport channel allows a socket to remain idle between requests.

Data type	Integer
Default	30 seconds

Enable access and error logging

When selected, the HTTP transport channel performs NCSA access and error logging. Enabling NCSA access and error logging slows server performance.

To configure NCSA access and error logging, click **HTTP error and NCSA access logging** under **Related Items**. Even if HTTP error and NCSA access logging is configured, it is not enabled unless the Enable access and error logging property is selected.

The default value for the Enable access and error logging property is not selected.

TCP transport channel settings

Use this page to view and configure a TCP transport channels. This type of transport channel handles inbound TCP/IP requests from a remote client.

To view this administrative console page, click **Servers** > **Application servers** > *server_name* > **Ports** > . Click on **View associated transports** for the port associated with the TCP transport channel whose settings you want to view.

Transport channel name

Specifies the name of the TCP transport 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

Port

Specifies the TCP/IP port this transport channel uses to establish connections between a client and an application server. The TCP transport channel binds to the hostnames and ports listed for the Port property. You can specify the wildcard * (an asterisk), for the hostname if you want this channel to listen to all hosts that are available on this system. However, before specifying the wildcard value, make sure this TCP transport channel does not have to bind to a specific hostname.

Data type	string

Maximum open connections

Specifies the maximum number of connections that can be open at one time.

Data type	Integer between 1 and 20,000 inclusive
Default	20,000

Inactivity timeout

Specifies the amount of time, in seconds, that the TCP transport channel waits for a read or write request to complete on a socket.

Note: The value specified for this property might be overridden by the wait times established for channels above this channel. For example, the wait time established for an HTTP transport channel overrides the value specified for this property for every operation except the initial read on a new socket.

Data type	Integer
Default	60 seconds

Address exclude list

Lists the IP addresses that are not allowed to make inbound connections.

Use a comma to separate the IPv4 or IPv6 or both addresses to which you want to deny access on inbound TCP connection requests.

All four numeric values in an IPv4 address must be represented by a number or the wildcard character * (an asterisk).

Following are examples of valid IPv4 addresses that can be included in an Address exclude list:

```
*.1.255.0
254.*.*.9
1.*.*.*
```

All eight numeric values of an IPv6 address must be represented by a number or the wildcard character * (an asterisk). No shortened version of the IPv6 address should be used. Even though a shortened version is processed with no error given, it does not function correctly in this list. Each numeric entry should be a 1- 4 digit hexadecimal number.

Following are examples of valid IPv6 addresses that can be included in an Address exclude list:

```
0:*:*:0:007F:0:0001:0001
F:FF:FFF:FFFF:1:01:001:0001
1234:*:4321:*:9F9f:*:*:0000
```

Note: The **Address include list** and **Host name include list** are processed before the **Address exclude list** and the **Host name exclude list**. If all four lists are defined:

- An address that is defined on either inclusion list will be allowed access provided it is not included on either of the exclusion lists.

- If an address is included in both an inclusion list and in an exclusion list, it will not be allowed access.

Address include list

Lists the IP addresses that are allowed to make inbound connections. Use a comma to separate the IPv4 or IPv6 or both addresses to which you want to grant access on inbound TCP connection requests.

All four numeric values in an IPv4 address must be represented by a number or the wildcard character * (an asterisk).

Following are examples of valid IP addresses that can be included in an Address include list:

```
 *.1.255.0
 254.*.*.9
 1.*.*.*
```

All eight numeric values of an IPv6 address must be represented by a number or the wildcard character * (an asterisk). No shortened version of the IPv6 address should be used. Even though a shortened version is processed with no error given, it does not function correctly in this list. Each numeric entry should be a 1- 4 digit hexadecimal number.

Following are examples of valid IPv6 addresses that can be included in an **Address include list**:

```
0:*:*:0:007F:0:0001:0001
F:FF:FFF:FFFF:1:01:001:0001
1234:*:4321:*:9F9f:*:*:0000
```

Note: The Address include list and the Host name include list are processed before the Address exclude list and the Host name exclude list. If all four lists are defined:

- An address that is defined on either inclusion list will be allowed access provided it is not included on either of the exclusion lists.
- If an address is included in both an inclusion list and in an exclusion list, it will not be allowed access.

Host name exclude list

List the host names that are not allowed to make connections. Use a comma to separate the URL addresses to which you want to deny access on inbound TCP connection requests.

A URL address can start with the wildcard character * (an asterisk) followed by a period; for example, `*.Rest.Of.Address`. If a period does not follow the wildcard character, the asterisk will be treated as a normal non-wildcard character. The wildcard character cannot appear any where else in the address. For example, `ibm.*.com` is not a valid hostname.

Following are examples of valid URL addresses that can be included in a Host name exclude list:

```
*.ibm.com
www.ibm.com
*.com
```

Note: The Address include list and Host name include list are processed before the Address exclude list and the Host name exclude list. If all four lists are defined:

- An address that is defined on either inclusion list will be allowed access provided it is not included on either of the exclusion lists.
- If an address is included in both an inclusion list and in an exclusion list, it is not allowed access.

Host name include list

Lists the host names that are allowed to make inbound connections. Use a comma to separate the URL addresses to which you want to grant access on inbound TCP connection requests.

A URL address can start with the wildcard character * (an asterisk) followed by a period; for example, `*.Rest.Of.Address`. If a period does not follow the wildcard character, the asterisk will be treated as a normal non-wildcard character. The wildcard character cannot appear any where else in the address. For example, `ibm.*.com` is not a valid hostname.

Following are examples of valid URL addresses that can be included in a hostname include list:

```
*.ibm.com
www.ibm.com
*.com
```

Note: The Address include list and Host name include list are processed before the Address exclude list and the Host name exclude list. If all four lists are defined:

- An address that is defined on either inclusion list will be allowed access provided it is not included on either of the exclusion lists.
- If an address is included in both an inclusion list and in an exclusion list, it is not allowed access.

DCS transport channel settings

Use this page to view and configure an DCS transport channels. This type of transport channel handles inbound Distribution and Consistency Services (DCS) messages.

By default, two channel transport chains are defined for an application server that contains a DCS channel:

- The chain named DCS contains a TCP and a DCS channel.
- The chain named DCS-Secure contains a TCP, an SSL, and a DCS channel.

Both of these chains terminate in, or use the same TCP channel instance. This TCP channel is associated with the DCS_UNICAST_ADDRESS port and is not used in any other transport chain. One instance of an SSL channel is reserved for use in the DCS-Secure chain. It also is not used in any other transport chains.

To view this administrative console page, click **Servers** > **Application servers** > *server_name* > **Ports** > . Click **View associated transports** for the port associated with the DCS transport channel whose settings you want to look at.

Transport channel name

Specifies the name of the DCS transport 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 DCS transport 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

ORB service transport channel settings

Use this page to view and configure an Object Request Broker (ORB) Service transport channels. This type of transport channel handles CORBA and RMI/IIOP inbound messages for the ORB Service. It enables clients to make requests and receive responses from servers in a network-distributed environment.

By default, two channel transport chains are defined for an application server that contains an ORB Service channel:

- The chain named ORB contains a TCP and an ORB Service channel.
- The chain named ORB-Secure contains a TCP, an SSL, and an ORB Service channel.

Both of these chains terminate in, or use the same TCP channel instance. This TCP channel is associated with the IIOP port and is not used in any other transport chain. One instance of an SSL channel is reserved for use in the DCS-Secure chain. It also is not used in any other transport chains.

To view this administrative console page, click **Servers > Application servers >** *server_name* **> Ports >** . Click on **View associated transports** for the port associated with the ORB Service transport channel whose settings you want to look at.

Transport channel name

Specifies the name of the ORB service transport 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 ORB service transport 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

SSL inbound channel

Use this page to determine which SSL inbound channel options to specify for the application server.

To view this administrative console page:

1. Click **Servers** > **Application Servers** > *server_name*.
2. Under Container settings, click **Web container transport chains** > *secure_transport_chain*.
3. Under Transport channels, click **SSL Inbound Channel (SSL_1)**.

Transport Channel Name

Specifies the name of the SSL inbound channel.

The name field cannot contain the following characters: **# \ / , : ; " * ? < > | = + & % '**

This name must be unique across all channels in an application server environment. For example, an SSL 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

Centrally managed

Specifies that the selection of an SSL configuration is based upon the outbound topology view for the Java Naming and Directory Interface (JNDI) platform.

Centrally managed configurations support one location to maintain SSL configurations rather than spreading them across the configuration documents.

Default:	Enabled

Specific to this endpoint

Specifies the SSL configuration alias that you want to use for outbound SSL communications.

This option overrides the centrally managed configuration for the JNDI (LDAP) protocol.

Session Initiation Protocol (SIP) inbound channel settings

Use this page to configure the SIP inbound channel settings.

To view this administrative console page, click **Servers** > **Application servers** > *server_name* > **Ports** . Click on **View associated transports** for the port associated with the UDP transport channel whose settings you want to view.

Transport channel name

Specifies the name of the SIP inbound transport 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 transport channel and a TCP transport channel cannot have the same name if they reside within the same system.

Default	UDP_(n) where (n) represents the number of instances of this channel in the system

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	10

Session Initiation Protocol (SIP) container inbound channel settings

Use this page to configure the SIP container inbound channel settings.

To view this administrative console page, click **Servers** > **Application servers** > *server_name* > **Ports** . Click on **View associated transports** for the port associated with the UDP transport channel whose settings you want to view.

Transport channel name

Specifies the name of the SIP container inbound transport 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 container transport channel and a TCP transport channel cannot have the same name if they reside within the same system.

Default	UDP_(n) where (n) represents the number of instances of this channel in the system

Descrimination 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	10

User Datagram Protocol (UDP) Inbound channel settings

Use this page to configure the UDP Inbound channel settings.

To view this administrative console page, click **Servers** > **Application servers** > *server_name* > **Ports** . Click on **View associated transports** for the port associated with the UDP transport channel whose settings you want to view.

Transport channel name

Specifies the name of the UDP inbound transport 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 UDP transport channel and a TCP transport channel cannot have the same name if they reside within the same system.

Default	UDP_(n) where (n) represents the number of instances of this channel in the system

Address exclude list

Specifies the IP addresses that are not allowed to make inbound connections. Use a comma to separate the IPv4 and/or IPv6 addresses to which you want to deny access on inbound UDP connection requests.

The address include list and host name include list are processed before the address exclude list and the host name exclude list. If all four lists are defined:

- An address that is defined on either inclusion list will be allowed access provided it is not included on either of the exclusion lists.
- If an address is included in both an inclusion list and in an exclusion list, it is not allowed access.

Data type
String

Range
Valid IPv4 and IPv6 addresses with a wildcard character (*), an asterisk. All four elements of an IPv4 address must be represented by a number or a wildcard character. All eight numeric values of an IPv6 address must be represented by a number or the wildcard character (*).

Example
The following examples are valid IPv4 addresses that can be included in an Address exclude list:

```
*.1.255.0
254.*.*.9
1.*.*.*
```

All eight numeric values of an IPv6 address must be represented by a number or the wildcard character (*), an asterisk. No shortened version of the IPv6 address should be used. Even though a shortened version is processed with no error given, it does not function correctly in this list. Each numeric entry should be a 1- 4 digit hexadecimal number. The following examples are valid IPv6 addresses that can be included in an Address exclude list:

```
0:*:*:0:007F:0:0001:0001
F:FF:FFF:FFFF:1:01:001:0001
1234:*:4321:*:9F9f:*:*:0000
```

Address include list

Specifies the IP addresses that are allowed to make inbound connections. Use a comma to separate the IPv4 and/or IPv6 addresses to which you want to allow access on inbound UDP connection requests.

Data type
String

Range
Valid IPv4 and IPv6 addresses with a wildcard character (*), an asterisk. All four elements of an IPv4 address must be represented by a number or a wildcard character (*). All eight numeric values of an IPv6 address must be represented by a number or the wildcard character (*).

Web container inbound transport channel settings

Use this page to view and configure a Web container inbound channel transport. This type of channel transport handles inbound Web container requests from a remote client.

To view this administrative console page, click **Servers** > **Application servers** > *server_instance* > **Container Settings** > **Web Container Settings**> **Web container transport chains** > *transport_chain* > **Web container inbound channel (***transport_channel_name***)** .

Transport Channel Name

Specifies the name of the Web container inbound transport 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 Web container inbound transport 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

Write buffer size

Specifies the amount of content in bytes to buffer unless the servlet explicitly calls flush/close on the response/writer output stream.

Data type	bytes
Default	9192 bytes

Developing custom services

A custom service provides the ability to plug into a WebSphere Application Server application server to define a hook point that runs when the server starts and shuts down. An application server configuration provides settings that control how an application server provides services for running applications and their components.

A developer implements a custom service containing a class that implements a particular interface. The administrator configures the custom service in the administrative console, identifying the class created by the developer. When an application server starts, any custom services defined for the application server are loaded and the server runtime calls their initialize methods.

Custom services run in servants, not in controllers. For example, because there can be more than one servant started in the life of a server and these servants can be started long after the server (controller) is up, as needed by WLM, a custom service runs during the start of each servant.

To define a hook point to be run when a server starts and shuts down, you develop a custom service class and then use the administrative console to configure a custom service instance. When the application server starts, the custom service starts and initializes.

The following restrictions apply to the WebSphere Application Server custom services implementation:

- The init and shutdown methods must return control to the runtime.
- No work is dispatched into the server instance until all custom service initialize methods return.
- The init and shutdown methods are called only once on each service, and once for each operating system process that makes up the server instance. File I/O is supported.
- Initialization of process level static data, without leaving the process, is supported.
- Only JDBC RMLT (resource manager local transaction) operations are supported. Every unit of work (UOW) must be completed before the methods return.
- Creation of threads is not supported.
- Creation of sockets and I/O, other than file I/O, is not supported. Running standard J2EE code, such as client code, servlets, and enterprise beans, is not supported.

- The Java Transaction API (JTA) interface is not available. This feature is available in J2EE server processes and distributed generic server processes only.
- While the runtime makes an effort to call shutdown, there is no guarantee that shutdown will be called prior to process termination.

These restrictions apply to the shutdown and init methods equally. Some JNDI operations are available.

1. Develop a custom service class that implements the com.ibm.websphere.runtime.CustomService interface. The properties passed by the application server runtime to the initialize method can include one for an external file containing configuration information for the service. You can use the externalConfigURLKey property to retrieve this information. In addition, these properties can contain any name-value pairs that are stored for the service, along with the other system administration configuration data for the service. The properties are passed to the initialize method of the service as a Properties object.

   There is a shutdown method for the interface as well. Both methods of the interface declare that they may create an exception, although no specific exception subclass is defined. If an exception is created, the runtime logs it, disables the custom service, and proceeds with starting the server.
2. Configure the custom service.

   In the administrative console, click **Servers > Application Servers**, and then under Server Infrastructure, click **Custom Services > New**. Then, on the settings page for a custom service instance, create a custom service configuration for an existing application server, supplying the name of the class implemented. If your custom service class requires a configuration file, specify the fully-qualified path name to that configuration file in the **externalConfigURL** field. This file name is passed into your custom service class.

   To invoke a native library from the custom service, provide the path name in the **Classpath** field in addition to the path names that are used to locate the classes and JAR files for the custom service. Doing this adds the path name to the WebSphere Application Server extension classloader, which allows the custom service to locate and correctly load the native library.
3. Stop the application server and restart it.

   Stop the application server and then restart the server.
4. Ensure the initialize and shutdown methods of the custom service perform as intended.

   Check the application server to ensure that the initialize method of the custom service ran as intended. Also ensure that the shutdown method performs as intended when the server or node agent stops.

As mentioned above, your custom services class must implement the CustomService interface. In addition, your class must implement the initialize and shutdown methods. Suppose the name of the class that implements your custom service is *ServerInit*, your code would declare this class as shown below. The code below assumes that your custom services class needs a configuration file. It shows how to process the input parameter in order to get the configuration file. If your class does not require a configuration file, the code that processes configProperties is not needed.

```
public class ServerInit implements CustomService
{
/**
* The initialize method is called by the application server run-time when the
* server starts. The Properties object passed to this method must contain all
* configuration information necessary for this service to initialize properly.
*
* @param configProperties java.util.Properties
*/
    static final java.lang.String externalConfigURLKey =
       "com.ibm.websphere.runtime.CustomService.externalConfigURLKey";

    static String ConfigFileName="";

    public void initialize(java.util.Properties configProperties) throws Exception
    {
        if (configProperties.getProperty(externalConfigURLKey) != null)
        {
           ConfigFileName = configProperties.getProperty(externalConfigURLKey);
        }

       // Implement rest of initialize method
    }
/**
* The shutdown method is called by the application server run-time when the
* server begins its shutdown processing.
*
* @param configProperties java.util.Properties
*/
    public void shutdown() throws Exception
    {
        // Implement shutdown method
    }
```

Custom service collection

Use this page to view a list of services available to the application server and to see whether the services are enabled. A custom service provides the ability to plug into a WebSphere application server and define code that runs when the server starts or shuts down.

To view this administrative console page, click **Servers > Application servers >** *server_name*. Then, under Server Infrastructure, click **Administration > Custom Services**.

External Configuration URL

Specifies the URL for a custom service configuration file.

If your custom services class requires a configuration file, the value provides a fully-qualified path name to that configuration file. This file name is passed into your custom service class.

Classname

Specifies the class name of the service implementation. This class must implement the Custom Service interface.

Display Name

Specifies the name of the service.

Enable service at server startup

Specifies whether the server attempts to start and initialize the service when its containing process (the server) starts. By default, the service is not enabled when its containing process starts.

Custom service settings

Use this page to configure a service that runs in an application server.

To view this administrative console page, click **Servers > Application servers >** *server_name*. Then, under Server Infrastructure, click **Administration > Custom services >***custom_service_name*.

Enable service at server startup:

Specifies whether the server attempts to start and initialize the service when its containing process (the server) starts. By default, the service is not enabled when its containing process starts.

Data type	Boolean
Default	false

External Configuration URL:

Specifies the URL for a custom service configuration file.

If your custom services class requires a configuration file, specify the fully-qualified path name to that configuration file for the value. This file name is passed into your custom service class.

Data type	String
Units	URL

Classname:

Specifies the class name of the service implementation. This class must implement the Custom Service interface.

Data type	String
Units	Java class name

Display Name:

Specifies the name of the service.

Data type	String

Description:

Describes the custom service.

Data type	String

Classpath:

Specifies the class path used to locate the classes and JAR files for this service.

Data type	String
Units	Class path

Defining application server processes

To enhance the operation of an application server, you can define command-line information for starting or initializing an application server process. Such settings define runtime properties such as the program to run, arguments to run the program, and the working directory.

A process definition can include characteristics such as Java virtual machine (JVM) settings, standard in, error and output paths, and the user ID and password under which a server runs.

1. In the administrative console, click **Servers > Application Servers** click on an application server name, and then click **Java and Process Management > Process Definition**.

   You can also define application server processes using the wsadmin tool. For more information, see the *Using the administrative clients* PDF.
2. On the settings page for a process definition, specify the name of the executable to run, any arguments to pass when the process starts running, and the working directory in which the process will run. Then click **OK**.
3. Specify process execution statementsfor starting or initializing a UNIX or i5/OS process.
4. Specify monitoring policies to track the performance of a process.

5. Specify process logs to which standard out and standard error streams write. Complete this step if you do not want to use the default file names.
6. Specify name-value pairs for properties needed by the process definition.

   Important: Each custom property name must be unique. If the same name is used for multiple properties, the process uses the value specified for the first property that has that name.
7. Stop the application server and then restart the server.
8. Check the application server to ensure that the process definition runs and operates as intended.

Process definition settings

Use this page to configure a process definition. A process definition includes the command line information necessary to start or initialize a process.

To view this administrative console page, click **Servers > Application Servers >** *server_name*. Then under Server Infrastructure click **Java Process Management > Process Definition**.

For the z/OS platform, this page provides command-line information for starting, initializing, or stopping a process. Each of the commands for which information is provided can be used for the control process. Only the Start command and Start Command Args information applies for the servant process. Specify the commands for the control process on one process definition panel and the commands for the servant process on another process definition panel. Do not specify the commands for the two different processes on the same panel.

Executable Name

This command line information specifies the executable name that is invoked to start the process.

This field is not available for the z/OS control process.

Data type	String

Executable Arguments

This command line information specifies the arguments that are passed *arg1 arg2 arg3*.

This field is not available for the z/OS control process.

Data type	String
Units	Java command-line arguments

Start Command (startCommand)

This command line information specifies the platform-specific command to launch the server process.
z/OS control process

Data type	String
Format	START *control_JCL_procedure_name*
Example	START BBO6ACR

z/OS servant process

> For the z/OS servant process, the value on the start command specifies the procedure name that workload manager (WLM) uses to start the servant process. WLM only uses this value if the WLM dynamic application environment feature is installed.

Data type	String
Format	*servant_JCL_procedure_name*
Example	BBO6ASR

Start Command Args (startCommandArgs)

This command line information specifies any additional arguments required by the start command.

z/OS control process

Data type	String
Format	JOBNAME=*server_short_name*,ENV=*cell_short_name.node_short_name.serv*
Example	JOBNAME=BBOS001,ENV=SY1.SY1.BBOS001

z/OS servant process

Data type	String
Format	JOBNAME=*server_short_name*S,ENV=*cell_short_name.node_short_name.se*
Example	JOBNAME=BBOS001S,ENV=SY1.SY1.BBOS001

Note: For the z/OS platform, the server short name (JOBNAME) contains 7 characters by default, but you can lengthen the short name to 8 characters.

Stop Command (stopCommand)

This command line information specifies the platform-specific command to stop the server process

Specify two commands in the field, one for the Stop command and one for the Immediate Stop (CANCEL) command.

Data type	String
Format	STOP *server_short_name*;CANCEL *server_short_name*
z/OS example	STOP BBOS001;CANCEL BBOS001

Stop Command Args (stopCommandArgs)

This command line information specifies any additional arguments required by the stop command.

Specify arguments for the Stop command and the Immediate Stop (CANCEL) command.

Data type	String
Format	*stop command arg string*;*immediate stop command arg string*
Example	;ARMRESTART In this example, Stop has no arguments. Immediate Stop has the argument ARMRESTART. A semicolon precedes ARMRESTART.

Terminate Command (terminateCommand)

This command line information specifies the platform-specific command to terminate the server process.

Data type	String
Format	FORCE *server_short_name*
z/OS example	FORCE BBOS001

Terminate Command Args (terminateCommandArgs)

This command line information specifies any additional arguments required by the terminate command.

The default is an empty string.

Data type	String
Format	*terminate command arg string*
z/OS example	ARMRESTART

Working directory

Specifies the file system directory that the process uses as its current working directory. The process uses this directory to determine the locations of input and output files with relative path names.

This field is not available for the z/OS control process.

Data type	String

Executable target type

Select whether the executable target is a Java class or an executable JAR file.

Executable target

Specifies the name of the executable target. If the target type is a Java class name, this field contains the main() method. If the target type is an executable JAR file, this field contains the name of that JAR file.

Data type	String

Process execution settings

Use this page to view or change the process execution settings for a server process that applies to either an application server, a node agent or a deployment manager.

To view this administrative console page for an application server, click **Servers > Application Servers >** *server_name*. Then, under Server Infrastructure, click **Java and Process Management > Process Execution**.

To view this administrative console page for a node agent, click **System Administration > Node agents >***node_agent_name*. Then, under Server Infrastructure , click **Java and Process Management > Process Definition > Process Execution**.

To view this administrative console page for a deployment manager, click **System Administration > Deployment manager**. Then, under Server Infrastructure, click **Java and Process Management > Process Definition > Process Execution**.

Process Priority:

Specifies the operating system priority for the process. The administrative process that launches the server must have root operating system authority in order to honor this setting.

Data type	Integer
Default	20 for WebSphere Application Server on all operating systems.

UMASK:

Specifies the user mask under which the process runs (the file-mode permission mask).

Data type	Integer

Run As User:

Specifies the user that the process runs as.

Data type	String

Run As Group:

Specifies the group that the process is a member of and runs as.

On i5/OS, the Run As Group setting is ignored.

Data type	String

Run In Process Group:

Specifies a specific process group for the process. A process group is a mechanism that the operating system uses to logically associate multiple processes and operate on them as a single unit. Usually, the operating system uses this mechanism for signal distribution.

Specific operating systems might allow other operations to be performed on a process group. Refer to your operating system documentation for more information on the operations that can be performed on a process group.

Data type	Integer
Default	0, which indicates that the process is not assigned to a specific process group.

Process logs settings

Use this page to view or change settings for specifying the files to which standard out and standard error streams write.

To view this administrative console page, in the administrative console:

For an application server on the z/OS platform, click **Servers > Application Servers >** *server_name*, and then, under Server Infrastructure, click **Java and Process Management > Process Definition > Servant > Process Logs**.
For a deployment manager on the z/OS platform, click **System Administration > Deployment Manager**, and then under Server Infrastructure, click **Java and Process Management > Process Definition > Servant > Process Log**.

Stdout File Name:

Specifies the file to which the standard output stream is directed. The file name can include a symbolic path name defined in the variable entries.

Use the field on the configuration tab to specify the file name. Use the field on the Runtime tab to select a file for viewing. View the file by clicking **View**.

Direct server output to the administrative console or to the process that launched the server, by either deleting the file name or specifying `console` on the configuration tab.

Data type	String
Units	File path name

Stderr File Name:

Specifies the file to which the standard error stream is directed. The file name can include a symbolic path name defined in the variable entries.

Use the field on the configuration tab to specify the file name. Use the field on the runtime tab to select a file for viewing. View the file by clicking **View**.

Data type	String
Units	File path name

Monitoring policy settings

Use this page to view or change settings that control how the node agent monitors and restarts a process.

To view this administrative console page, click **Servers > Application Servers >** *server_name*. Then, under Server Infrastructure, click **Java and Process Management > Monitoring Policy**.

Maximum Startup Attempts:

Specifies the maximum number of times to attempt to start the application server before giving up.

Data type	Integer

Ping Interval:

Specifies the frequency of communication attempts between the parent process, such as the node agent, and the process it has spawned, such as an application server. Adjust this value based on your requirements for restarting failed servers. Decreasing the value detects failures sooner; increasing the value reduces the frequency of pings, reducing system overhead.

Data type	Integer
Range	Set the value greater than or equal to 0 (zero) and less than 2147483647. If you specify a value greater than 2147483647, the application server acts as though you set the value to 0.

Ping Timeout:

When a parent process is spawning a child process, such as when a process manager spawns a server, the parent process pings the child process to see whether the child was spawned successfully. This value specifies the number of seconds that the parent process should wait (after pinging the child process) before assuming that the child process failed.

Data type	Integer
Units	Seconds
Range	Set the value greater than or equal to 0 (zero) and less than 2147483647. If you specify a value greater than 2147483647, the application server acts as though you set the value to 0.

Automatic Restart:

Specifies whether the process should restart automatically if it fails. On distributed systems, the default is to restart the process automatically. On a z/OS system, the default is to not start the process automatically.

If you change the value specified for this field, you must restart the application server and the node agent before the new setting takes effect.

This setting does not affect what you specified for the Node Restart State setting. The two settings are mutually exclusive.

Data type	Boolean
Default	true (distributed) / false (z/OS)

Node Restart State:

Specifies the desired behavior of the servers after the node completely shuts down and restarts.

If a server is already running when the node agent stops, that server is still running after the node agent restarts. If a server is stopped when the node agent restarts, whether the node agent starts the server depends on the setting for this property:

- If this property is set to STOPPED, node agent does not start the server.
- If this property is set to RUNNING, the node agent always starts the server.
- If this property is set to PREVIOUS, the node agent starts the server only if the server was running when the node agent stopped.

This setting does not affect what you specified for the Automatic Restart setting. The two settings are mutually exclusive.

Data type	String
Default	STOPPED
Range	Valid values are STOPPED, RUNNING, or PREVIOUS. If you want the process to return to its current state after the node restarts, use PREVIOUS.

Process definition type settings

Use this page to view or change settings for a process definition type. This page only displays for the z/OS platform.

To view this administrative console page, click **Servers > Application Servers >** *server_name* **> Process Definition**.

Control:

Specifies the process definitions for the control process

Servant:

Specifies the process definitions for the servant process.

Sysplex Distributor

The IBM-recommended implementation, if you are running in a sysplex, is to set up your TCP/IP network with Sysplex Distributor. This makes use of dynamic virtual IP addresses (DVIPAs), which increase availability and aid in workload balancing.

The following are recommended environment considerations for Sysplex Distributor:

- You need only basic sysplex functionality to utilize DVIPAs and Sysplex Distributor because these functions do not rely on data stored permanently in the coupling facility.

- Set up your system such that each HTTP request connection results in no saved state or the HTTP and application servers are configured to share a persistent state.

When doing this, HTTP server plug-ins send no-affinity connections to Sysplex Distributor (a secondary connection load balancer) with more information to make a better distribution decision.

Note: As long as the HTTP catcher itself is not bound to any particular IP address, the application-specific DVIPA can be used when affinities dictate a particular server. This allows use of the Sysplex Distributor server address for requests that are not tied to a server, covering the same set of servers in the sysplex.

Since the client connection terminates at the plug-in/proxy, and the secondary connection is established by the plug-in itself, there is no need for network address translation.

Requests to the node agent do not require any affinity, and each request is independent of other requests. Sysplex Distributor can be used to balance work requests among node agents, with the added benefit that Sysplex Distributor knows which nodes are available. Therefore, it will never route a work request to a node that is not listening for new connection requests.

Note: If you are running z/OS 1.2 or earlier, Sysplex Distributor is limited to distribution on only four ports for a particular distributed DVIPA. You may configure multiple DVIPAs when more than four ports exist, but this is a configuration burden.

Running multiple TCP/IP stacks

You might want to run multiple TCP/IP stacks on the same system to provide network isolation for one or more of your applications. For instance, you may have multiple Overflow Sequential Access (OSA) features, each one connecting your system to a different network. You can assign a TCP/IP stack to each feature.

When configuring WebSphere Application Server for z/OS on a system with multiple stacks, you must first establish the Application Server's stack affinity to the desired stack so that all socket communications are bound to that stack, and then you establish the Application Server's allocation of the proper host name resolution configuration data sets so that host name lookups have the desired results.

Use the NETWORK DOMAINNAME parameter of SYS1.PARMLIB(BPXPRMxx) to specify the common INET physical file system, C_INET PFS, and then use this file system to set up multiple TCP/IP stacks. This physical file system allows you to configure multiple physical file systems (network sockets) and make them active concurrently.

If you plan to configure WebSphere Application Server for z/OS to use a non-default TCP/IP stack, consult *z/OS UNIX System Services Planning*, and *z/OS Communications Server: IP Configuration Reference*, for details.

Note: In the steps below, you will set a number variables. It is important to understand that these variables should be set at the node level.

To configure WebSphere Application Server for z/OS on a system with multiple stacks:

1. Configure the data set for each Application Server's host name resolution. In the administrative console, click **Environment** > **WebSphere Variables** > **New**.
   a. Add the RESOLVER_CONFIG UNIX process variable and specify the data set name in the **value** field.
   b. Export the RESOLVER_CONFIG variable in client shell scripts.
   - You can also use JCL to specify the name resolution configuration data set. To use JCL, add //SYSTCPD DD DSN=some.tcpip.DATA,DISP=SHR to the server JCL. The RESOLVER_CONFIG variable overrides the SYSTCPD DD statement.

See *z/OS Communications Server: IP Configuration Reference*, for more information on the RESOLVER_CONFIG variable.

2. Establish the Application Server's stack affinity to the desired stack.
   a. In the administrative console, click **Environment** > **WebSphere Variables** and set the _BPXK_SETIBMOPT_TRANSPORT UNIX process variable to the value of the desired transport. If this variable does not exist, click **New** and add it.
   b. Export the _BPXK_SETIBMOPT_TRANSPORT variable in client shell scripts.

   See *z/OS UNIX System Services Planning*, for more information on the _BPXK_SETIBMOPT_TRANSPORT variable.

Bind-specific support in WebSphere Application Server for z/OS

Bind-specific support in WebSphere Application Server for z/OS allows you to control the use of TCP/IP resources in WebSphere Application Server for z/OS.

This support allows you to have the Application Server ORB and other products and applications on the same z/OS system without requiring the client code to configure unique ports. In other words, this support allows use of port 2809 by the Application Server and other products and applications on the same system. This support allows the utilization of multiple TCP/IP stacks (Common INET) by the WebSphere for z/OS ORB and the use of multiple IP addresses on the same TCP/IP stack.

To use bind-specific support, use the **ORB_LISTENER_ADDRESS** end point, which specifies the IP address in dotted decimal format. WebSphere Application Server for z/OS servers listen for client connection requests on this IP address. See "Ports settings" on page 258 for more information.

Because a given IP address is associated with a given TCP/IP stack, you can specify the ORB_LISTENER_ADDRESS endpoint so that WebSphere Application Server for z/OS servers use a specific TCP/IP stack.

In addition, because you can define multiple IP addresses for a given TCP/IP stack, the Application Server port 2809 servers could share the same TCP/IP stack with other products and applications requiring port 2809, because you made their IP addresses unique with the ORB_LISTENER_ADDRESS end point.

Alternatively, you can, without the use of bind-specific support, define alternate ports for port 2809 and the location service daemon, which are the only values defined by the CORBA standard. However it is not clear that all client ORBs will easily support configuring the Application Server port to something other than 2809. Configure the ports for the location service daemon and node by specifying port numbers on the z/OS location service daemon settings page in the administrative console.

For more information about multiple TCP/IP stacks (Common INET), see *z/OS UNIX System Services Planning*. For more information about multiple IP addresses on the same TCP/IP stack, see *z/OS Communications Server: IP Configuration Reference*.

Configuring the JVM

As part of configuring an application server, you might define settings that enhance the way your operating system uses of the Java virtual machine (JVM).

The Java virtual machine (JVM) is an interpretive computing engine responsible for running the byte codes in a compiled Java program. The JVM translates the Java byte codes into the native instructions of the host machine. The application server, being a Java process, requires a JVM in order to run, and to support the Java applications running on it. JVM settings are part of an application server configuration.

To view and change the JVM configuration for an application server's process, use the Java virtual machine page of the administrative console or use wsadmin to change the configuration through scripting.

1. In the administrative console, click **Servers > Application Servers >***server***> Java and Process Management > Process Definition > Java Virtual Machine**.
2. Specify values for the JVM settings as needed and click **OK**.
3. Click **Save** on the console task bar.
4. Restart the application server.

"Configuring application servers for UCS Transformation Format" on page 252 provides an example that involves specifying a value for the **Generic JVM Arguments** property on the Java virtual machine page to enable UTF-8 encoding on an application server. Enabling UTF-8 allows multiple language encoding support to be used in the administrative console.

"Configuring JVM sendRedirect calls to use context root" on page 355 provides an example that involves defining a property for the JVM.

Java virtual machine settings

Use this page to view and change the Java virtual machine (JVM) configuration settings of a process for an application server.

To view this administrative console page, connect to the administrative console and navigate to the Java virtual machine panel:

For the z/OS platform:

Application server	**Servers** > **Application Servers** > *server1* > **Process Definition** > *Control* > **Java Virtual Machine**
Deployment manager	**System Administration** > **Deployment Manager** > **Process definition** > *Control* > **Java Virtual Machine**
Node agent	**System Administration** >**Node Agent** > *nodeagent* > **Process definition** >*Control* > **Java Virtual Machine**

Classpath

Specifies the standard class path in which the Java virtual machine code looks for classes.

Enter each classpath entry into a table row. You do not need to add the colon or semicolon at the end of each entry.

Data type	String
Units	Class path

Boot classpath

Specifies bootstrap classes and resources for JVM code. This option is only available for JVM instructions that support bootstrap classes and resources. You can separate multiple paths by a colon (:) or semi-colon (;), depending on operating system of the node.

Data type	String

Verbose class loading

Specifies whether to use verbose debug output for class loading. The default is not to enable verbose class loading.

Data type	Boolean
Default	false

Verbose garbage collection

Specifies whether to use verbose debug output for garbage collection. The default is not to enable verbose garbage collection.

Data type	Boolean
Default	false

Verbose JNI

Specifies whether to use verbose debug output for native method invocation. The default is not to enable verbose Java Native Interface (JNI) activity.

Data type	Boolean
Default	false

Initial heap size

Specifies the initial heap size available to the JVM code, in megabytes.

Increasing the minimum heap size can improve startup. The number of garbage collection occurrences are reduced and a 10% gain in performance is realized.

Increasing the size of the Java heap improves throughput until the heap no longer resides in physical memory, in general. After the heap begins swapping to disk, Java performance suffers drastically.

Data type	Integer
Default	For z/OS, the default for the controller is 48, and the default for the servant is 128.

Maximum heap size

Specifies the maximum heap size available to the JVM code, in megabytes.

Increasing the heap size can improve startup. By increasing heap size, you can reduce the number of garbage collection occurrences with a 10% gain in performance.

Increasing the size of the Java heap usually improves throughput until the heap no longer resides in physical memory. When the heap size exceeds the physical memory, the heap begins swapping to disk which causes Java performance to drastically decrease. Therefore, it is important to set the maximum heap size to a value that allows the heap to be contained within physical memory.

To prevent paging, you should allow a minimum of 256MB of physical memory for each processor and 512 MB of physical memory for each application server. If possible, adjust the available memory when paging occurs if processor utilization is low because of this paging.

Data type	Integer
Default	For z/OS and distributed platforms, the default is 256. Keep the value low enough to avoid paging or swapping-out-memory-to-disk.

Debug mode

Specifies whether to run the JVM in debug mode. The default is not to enable debug mode support.

If you set the Debug Mode property to true, then you must specify command-line debug arguments as values for the Debug Arguments property.

Data type	Boolean

Default	false

Debug arguments

Specifies command-line debug arguments to pass to the JVM code that starts the application server process. You can specify arguments when Debug Mode is enabled.

For WebSphere Application Server Network Deployment configurations, Debug arguments are only required if the Debug Mode property is set to true. If you enable debugging on multiple application servers on the same node, make sure that the servers are using different `address` arguments, which define the port for debugging. For example, if you enable debugging on two servers and leave the default debug port for each server as `address=7777`, the servers could fail to start properly.

Data type	String
Units	Java command-line arguments

Generic JVM arguments

Specifies command line arguments to pass to the Java virtual machine code that starts the application server process.

The following are optional command line arguments that you can ente int the Generic JVM arguments field. If you enter more than one argument, enter a space between each argument.

Important: If the argument says it is for the IBM Developer Kit only, you cannot use the argument with another JVM, such as the Sun JDK or the HP JDK.

- **-Xquickstart**

  You can use **-Xquickstart** for initial compilation at a lower optimization level than in default mode. Later, depending on sampling results, you can recompile to the level of the initial compile in default mode. Use **-Xquickstart** for applications where early moderate speed is more important than long run throughput. In some debug scenarios, test harnesses and short-running tools, you can improve startup time between 15-20%.
- **-Xverify:none**

  You can use **-Xverify:none** if you want to skip the class verification stage during class loading . Using **-Xverify:none** with the just in time (JIT) compiler enabled, improves startup time by 10-15%.
- **-Xnoclassgc**

  You can use **-Xnoclassgc** to disable class garbage collection. This action leads to more class reuse and slightly improved performance. The trade-off is that you won't be collecting the resources owned by these classes. You can monitor garbage collection using the `verbose:gc` configuration setting, which will output class garbage collection statistics. Examining these statistics will help you understand the trade-off between the reclaimed resources and the amount of garbage collection required to reclaim the resources. However, if the same set of classes are garbage collected repeatedly in your workload, you should disable garbage collection. Class garbage collection is enabled by default.
- **-Xgcthreads**

  You can use several garbage collection threads at one time, also known as *parallel garbage collection*. When entering this value in the Generic JVM arguments field, also enter the number of processors that your machine has, for example, **-Xgcthreads***n*, where *n* is the number of processors. On a node with *n* processors, the default number of threads is *n*. You should use parallel garbage collection if your machine has more than one processor. This argument is valid only for the IBM Developer Kit.
- **-Xnocompactgc**

  You can use **-Xnocompactgc** to disable heap compaction, which is the most expensive garbage collection operation. Avoid compaction in the IBM Developer Kit. If you disable heap compaction, you eliminate all associated overhead.
- **-Xinitsh**

You can use **-Xinitsh** to set the initial heap size where class objects are stored. The method definitions and static fields are also stored with the class objects. Although the system heap size has no upper bound, set the initial size so that you do not incur the cost of expanding the system heap size, which involves calls to the operating system memory manager. You can compute a good initial system heap size by knowing the number of classes loaded in the WebSphere Application Server product, which is about 8,000 classes, and their average size. Having knowledge of the applications helps you include them in the calculation. You can use this argument only with the IBM Developer Kit.

- **-Xgpolicy**

  You can use **-Xgpolicy** to set the garbage collection policy. If the garbage collection policy (gcpolicy) is set to `optavgpause`, concurrent marking is used to track application threads starting from the stack before the heap becomes full. The garbage collector pauses become uniform and long pauses are not apparent. The trade-off is reduced throughput because threads might have to do extra work. The default, recommended value is `optthruput`. Enter the value as `-Xgcpolicy:[optthruput|optavgpause]`. You can use this argument only with the IBM Developer Kit.

- **-XX**

  The Sun-based Java 2 Standard Edition (J2SE) 5 has generation garbage collection, which allows separate memory pools to contain objects with different ages. The garbage collection cycle collects the objects independently from one another depending on age. With additional parameters, you can set the size of the memory pools individually. To achieve better performance, set the size of the pool containing short lived objects so that objects in the pool do not live through more then one garbage collection cycle. The size of new generation pool is determined by the `NewSize` and `MaxNewSize` parameters.

  Objects that survive the first garbage collection cycle are transferred to another pool. The size of the survivor pool is determined by parameter `SurvivorRatio`. If garbage collection becomes a bottleneck, you can try customizing the generation pool settings. To monitor garbage collection statistics, use the object statistics in Tivoli Performance Viewer or the verbose:gc configuration setting. Enter the following values:

  ```
  -XX:NewSize (lower bound)
  -XX:MaxNewSize (upper bound)
   -XX:SurvivorRatio=NewRatioSize
  ```

  The default values are:`NewSize=2m MaxNewSize=32m SurvivorRatio=2`. However, if you have a JVM with more than 1 GB heap size, you should use the values: `-XX:newSize=640m -XX:MaxNewSize=640m -XX:SurvivorRatio=16`, or set 50 to 60% of total heap size to a new generation pool.

- **-Xminf**

  You can use **-Xminf** to specify the minimum free heap size percentage. The heap grows if the free space is below the specified amount. In reset enabled mode, this option specifies the minimum percentage of free space for the middleware and transient heaps. This is a floating point number, 0 through 1. The default is .3 (30%).

- **-server | -client**

  Java HotSpot Technology in the Sun-based Java 2 Standard Edition (J2SE) 5 uses an adaptive JVM containing algorithms for optimizing byte code execution over time. The JVM runs in two modes, **-server** and **-client**. If you use the default **-client** mode, there will be a faster startup time and a smaller memory footprint, but lower extended performance. You can enhance performance by using **-server** mode if a sufficient amount of time is allowed for the HotSpot JVM to warm up by performing continuous execution of byte code. In most cases, use **-server** mode, which produces more efficient run-time execution over extended periods. You can monitor the process size and the server startup time to check the difference between **-client** and **-server**.

- **-Dcom.ibm.CORBA.RequestTimeout=*timeout_interval***

  You can use **-Dcom.ibm.CORBA.RequestTimeout=*timeout_interval*** to specify the timeout period for responding to requests sent from the client. This argument uses the -D option. *timeout_interval* is the timeout period in seconds. If your network experiences extreme latency, specify a large value to prevent timeouts. If you specify a value that is too small, an application server that participates in workload management can time out before it receives a response.

  Be careful specifying this property; it has no recommended value. Set it only if your application is experiencing problems with timeouts.

- **-Dcom.ibm.websphere.wlm.unusable.interval=*interval***

  You can use the **-Dcom.ibm.websphere.wlm.unusable.interval=*interval*** argument to change the value for the com.ibm.websphere.wlm.unusable.interval property if the workload management state of the client is refreshing too soon or too late. This property specifies the time interval that the workload management client run time waits after it marks a server as unavailable before it attempts to contact the server again. This argument uses the -D option. *interval* is the time in seconds between attempts. The default value is 300 seconds. If the property is set to a large value, the server is marked as unavailable for a long period of time. This prevents the workload management refresh protocol from refreshing the workload management state of the client until after the time period has ended.
- **-Xshareclasses:none**

  You can use the **-Xshareclasses:none** argument to disable the share classes option for a process. The share classes option, which is available with Java 2 Standard Edition (J2SE) 5, lets you share classes in a cache. Sharing classes in a cache can improve startup time and reduce memory footprint. Processes, such as application servers, node agents, and deployment managers, can use the share classes option.

  If you use this option, you should clear the cache when the process is not in use. To clear the cache, either call the app_server_root/bin/clearClassCache.bat/sh utility or stop the process and then restart the process.

Data type	String
Units	Java command line arguments

Executable JAR file name

Specifies a full path name for an executable JAR file that the JVM code uses.

Data type	String
Units	Path name

Disable JIT

Specifies whether to disable the just in time (JIT) compiler option of the JVM code.

If you disable the JIT compiler, throughput decreases noticeably. Therefore, for performance reasons, keep JIT enabled.

Data type	Boolean
Default	false (JIT enabled)
Recommended	JIT enabled

Operating system name

Specifies JVM settings for a given operating system.

For the Network Deployment product, when the process starts, the process uses the JVM settings for the node as the JVM settings for the operating system.

Data type	String

Configuring JVM sendRedirect calls to use context root

If the com.ibm.websphere.sendredirect.compatibility property is not set and your application servlet code has statements such as *sendRedirect("/home.html")*, your Web browser might display messages such as *Error 404: No target servlet configured for uri: /home.html*.

transition: The com.ibm.websphere.sendredirect.compatibility property is deprecated. You should modify your applications to redirect non-relative URLs (those starting with a "/") relative to the servlet container (*web_server_root*) instead of relative to the Web application context root.

To instruct the server not to use the Web server's document root and to use instead the Web application's context root for sendRedirect() calls, configure the JVM by setting the com.ibm.websphere.sendredirect.compatibility property to a `true` or `false` value.

1. Access the settings page for a property of the JVM.
   a. Click **Servers > Application Servers** in the console navigation tree.
   b. On the Application Server page, click on the name of the server whose JVM settings you want to configure.
   c. On the settings page for the selected application server, under Server Infrastructure, click **Java and Process Management > Process Definition**.
   d. On the Process Definition page, click **Java Virtual Machine**.
   e. On the Java Virtual Machine page, click **Custom Properties**.
   f. On the Custom Properties page, click **New**.
2. On the settings page for a property, specify a name of `com.ibm.websphere.sendredirect.compatibility` and either `true` or `false` for the value, then click **OK**.
3. Click **Save** on the console task bar.
4. Stop the application server and then restart the application server.

Setting custom JVM properties

You can use the administrative console to change the values of JVM custom properties.

To set custom properties, connect to the administrative console and navigate to the Java virtual machine custom properties panel.

Application server	**Servers** > **Application Servers** >*server1* > **Process Definition** > *Control* > **Java Virtual Machine** > **Custom Properties**
Deployment manager	**System Administration** > **Deployment Manager** > **Process definition** > *Control* > **Java Virtual Machine** > **Custom Properties**
Node agent	**System Administration** >**Node Agent** > *nodeagent* > **Process definition** >*Control* > **Java Virtual Machine** > **Custom Properties**

If the custom property is not present in the list of already defined custom properties, create a new property, and enter the property name in the Name field and a valid value in the Value field. Restart the server to complete your changes.

To ensure the correct host is specified for the `End Point` property, navigate to the indicated page.

Application server	**Servers** > **Application Servers** >*server1*. Then, under Communications, click **Ports** > *port_name*
Deployment manager	**System Administration** > **Deployment Manager**. Then, under Additional Properties, click **Ports***port_name*

Node agent	**System Administration >Node Agent >** *nodeagent*. Then, under Additional Properties, click **Ports >** *port_name* **Important:** For node agents, there are MULTICAST ports NODE_MULTICAST_DISCOVERY_ADDRESS (IPv4) and NODE_MULTICAST_IPV6_DISCOVERY_ADDRESS (IPv6). This IP Address must be a CLASS D IP address, first octet is 224-239.

com.ibm.websphere.bean.delete.sleep.time

Specifies the time between sweeps to check for timed out beans. The value is entered in seconds. For example, a value of 120 would be 2 minutes. This property also controls the interval in the Servant process that checks for timed out beans still visible to the enterprise bean container.

The default value is 4200 (70 minutes). The minimum value is 60 (1 minute). The value can be changed through the administrative console. To apply this property, you must specify the value in both the Control and Servant JVM Custom Properties.

com.ibm.websphere.network.useMultiHome

In a multihomed environment where WebSphere Application Server is restricted to listen only on a specific IP address for Discovery and SOAP messages, set this property to false for the deployment manager, all Application Servers and all node agents. By default, the value of the property is true and the application server listens on all IP addresses on the host for Discovery and SOAP messages. If the property is set to false, WebSphere Application Server will only listen for Discovery and SOAP messages on the configured host name. If you set the property to false, you must also:

- Have a host name configured on WebSphere Application Server that resolves to a specific IP address.
- Ensure that the `end point` property for the deployment manager, all Application Servers, and all node agents is set to this host name. For the deployment manager, the end points that must be set are CELL_DISCOVERY_ADDRESS and SOAP_CONNECTOR_ADDRESS. For the node agent and application servers, only the SOAP_CONNECTOR_ADDRESS end point must be set.

You can change the value through the administrative console. Modify the defaults by setting the value for the server, deployment manager, and node agent. In order for these changes to take place, you must restart the server.

com.ibm.websphere.deletejspclasses

Deletes JavaServer Pages classes for all applications after those applications have been deleted or updated. By default, the value of this property is `true`.

com.ibm.websphere.deletejspclasses.delete

Deletes JavaServer Pages classes for all applications after those applications have been deleted, but not after they have been updated. By default, the value of this property is `true`.

com.ibm.websphere.deletejspclasses.update

Deletes JavaServer Pages classes for all applications after those applications have been updated, but not after they have been deleted. By default, the value of this property is `true`.

invocationCacheSize

The invocationCacheSize custom property is used to control the size of the invocation cache. The invocation cache holds information for mapping request URLs to servlet resources. A cache of the requested size is created for each worker thread that is available to process a request. The default size of the invocation cache is 50. If more than 50 unique URLs are actively being used (each JavaServer Page is a unique URL), you should increase the size of the invocation cache.

A larger cache uses more of the Java heap, so you might also need to increase the maximum Java heap size. For example, if each cache entry requires 2KB, maximum thread size is set to 25, and the URL invocation cache size is 100; then 5MB of Java heap are required.

You can specify any number higher than 0 for the cache size. Setting the value to zero disables the invocation cache.

Preparing to host applications

Rather than use the default application server provided with the product, you can configure a new server and set of resources.

The default application server and a set of default resources are available to help you begin quickly. You can choose instead to configure a new server and set of resources. Here is what you need to do in order to set up a runtime environment to support applications.

1. Create an application server.
2. Create a virtual host.
3. Configure a Web container. See the *Administering applications and their environment* PDF for more information.
4. Configure an EJB container. See the *Administering applications and their environment* PDF for more information.
5. Create resources for data access. See the *Administering applications and their environment* PDF for more information.
6. Create a JDBC provider and data source. See the *Administering applications and their environment* PDF for more information.
7. Create a URL and URL provider. See the *Administering applications and their environment* PDF for more information.
8. Create a JavaMail session. See the *Administering applications and their environment* PDF for more information.
9. Create resources for session support. See the *Administering applications and their environment* PDF for more information.
10. Configure a Session Manager. See the *Administering applications and their environment* PDF for more information.

Java memory tuning tips

Enterprise applications written in the Java language involve complex object relationships and utilize large numbers of objects. Although, the Java language automatically manages memory associated with object life cycles, understanding the application usage patterns for objects is important.

In particular, verify the following:
- The application is not over utilizing objects
- The application is not leaking objects
- The Java heap parameters are set properly to handle a given object usage pattern

Understanding the effect of garbage collection is necessary to apply these management techniques.

IBM Support has documents that can save you time gathering information needed to resolve this problem. Before opening a PMR, see the IBM Support page.

The garbage collection bottleneck

Examining Java garbage collection gives insight to how the application is utilizing memory. Garbage collection is a Java strength. By taking the burden of memory management away from the application writer, Java applications are more robust than applications written in languages that do not provide garbage collection. This robustness applies as long as the application is not abusing objects. Garbage collection normally consumes from 5% to 20% of total execution time of a properly functioning application. If not managed, garbage collection is one of the biggest bottlenecks for an application.

The Java virtual machine (JVM) uses a parallel garbage collector to fully exploit an SMP during most garbage collection cycles where the Sun HotSpot 1.3.1 JVM has a single-threaded garbage collector.

Monitoring garbage collection

You can use garbage collection to evaluate application performance health. By monitoring garbage collection during the execution of a fixed workload, you gain insight as to whether the application is over-utilizing objects. Garbage collection can even detect the presence of memory leaks.

For more information about monitoring garbage collection, see:

- The description of the DMPJVM command in the i5/OS Information Center. This command dumps JVM information for a specific job.

You can monitor garbage collection statistics using the **verbose:gc** JVM configuration setting. The **verbose:gc** format is not standardized between different JVMs or release levels.

For this type of investigation, set the minimum and maximum heap sizes to the same value. Choose a representative, repetitive workload that matches production usage as closely as possible, user errors included.

To ensure meaningful statistics, run the fixed workload until the application state is steady.

Detecting over-utilization of objects

You can check if the application is overusing objects, by observing the counters for the JVM runtime. You have to set the **-XrunpmiJvmpiProfiler** command line option, as well as the JVM module maximum level in order to enable the Java virtual machine profiler interface (JVMPI) counters. The best result for the average time between garbage collections is at least 5-6 times the average duration of a single garbage collection. If you do not achieve this number, the application is spending more than 15% of its time in garbage collection.

If the information indicates a garbage collection bottleneck, there are two ways to clear the bottleneck. The most cost-effective way to optimize the application is to implement object caches and pools. Use a Java profiler to determine which objects to target. If you can not optimize the application, adding memory, processors and clones might help. Additional memory allows each clone to maintain a reasonable heap size. Additional processors allow the clones to run in parallel.

Detecting memory leaks

Memory leaks in the Java language are a dangerous contributor to garbage collection bottlenecks. Memory leaks are more damaging than memory overuse, because a memory leak ultimately leads to system instability. Over time, garbage collection occurs more frequently until the heap is exhausted and the Java code fails with a fatal out-of-memory exception. Memory leaks occur when an unused object has references that are never freed. Memory leaks most commonly occur in collection classes, such as Hashtable because the table always has a reference to the object, even after real references are deleted.

High workload often causes applications to crash immediately after deployment in the production environment. This is especially true for leaking applications where the high workload accelerates the magnification of the leakage and a memory allocation failure occurs.

Memory leak testing

The goal of memory leak testing is to magnify numbers. Memory leaks are measured in terms of the amount of bytes or kilobytes that cannot be garbage collected. The delicate task is to differentiate these amounts between expected sizes of useful and unusable memory. This task is achieved more easily if the numbers are magnified, resulting in larger gaps and easier identification of inconsistencies. The following list contains important conclusions about memory leaks:

- **Long-running test**

  Memory leak problems can manifest only after a period of time, therefore, memory leaks are found easily during long-running tests. Short running tests can lead to false alarms. It is sometimes difficult to know when a memory leak is occurring in the Java language, especially when memory usage has seemingly increased either abruptly or monotonically in a given period of time. The reason it is hard to detect a memory leak is that these kinds of increases can be valid or might be the intention of the developer. You can learn how to differentiate the delayed use of objects from completely unused objects by running applications for a longer period of time. Long-running application testing gives you higher confidence for whether the delayed use of objects is actually occurring.
- **Repetitive test**

  In many cases, memory leak problems occur by successive repetitions of the same test case. The goal of memory leak testing is to establish a big gap between unusable memory and used memory in terms of their relative sizes. By repeating the same scenario over and over again, the gap is multiplied in a very progressive way. This testing helps if the number of leaks caused by the execution of a test case is so minimal that it is hardly noticeable in one run.

  You can use repetitive tests at the system level or module level. The advantage with modular testing is better control. When a module is designed to keep the private module without creating external side effects such as memory usage, testing for memory leaks is easier. First, the memory usage before running the module is recorded. Then, a fixed set of test cases are run repeatedly. At the end of the test run, the current memory usage is recorded and checked for significant changes. Remember, garbage collection must be suggested when recording the actual memory usage by inserting System.gc() in the module where you want garbage collection to occur, or using a profiling tool, to force the event to occur.
- **Concurrency test**

  Some memory leak problems can occur only when there are several threads running in the application. Unfortunately, synchronization points are very susceptible to memory leaks because of the added complication in the program logic. Careless programming can lead to kept or unreleased references. The incident of memory leaks is often facilitated or accelerated by increased concurrency in the system. The most common way to increase concurrency is to increase the number of clients in the test driver.

  Consider the following points when choosing which test cases to use for memory leak testing:
  - A good test case exercises areas of the application where objects are created. Most of the time, knowledge of the application is required. A description of the scenario can suggest creation of data spaces, such as adding a new record, creating an HTTP session, performing a transaction and searching a record.
  - Look at areas where collections of objects are used. Typically, memory leaks are composed of objects within the same class. Also, collection classes such as Vector and Hashtable are common places where references to objects are implicitly stored by calling corresponding insertion methods. For example, the get method of a Hashtable object does not remove its reference to the retrieved object.

Heap consumption indicating a possible leak during a heavy workload (the application server is consistently near 100% CPU utilization), yet appearing to recover during a subsequent lighter or near-idle workload, is an indication of heap fragmentation. Heap fragmentation can occur when the JVM can free sufficient objects to satisfy memory allocation requests during garbage collection cycles, but the JVM does not have the time to compact small free memory areas in the heap to larger contiguous spaces.

Another form of heap fragmentation occurs when small objects (less than 512 bytes) are freed. The objects are freed, but the storage is not recovered, resulting in memory fragmentation until a heap compaction has been run.

Heap fragmentation can be reduced by forcing compactions to occur, but there is a performance penalty for doing this. Use the Java **-X** command to see the list of memory options.

Java heap parameters

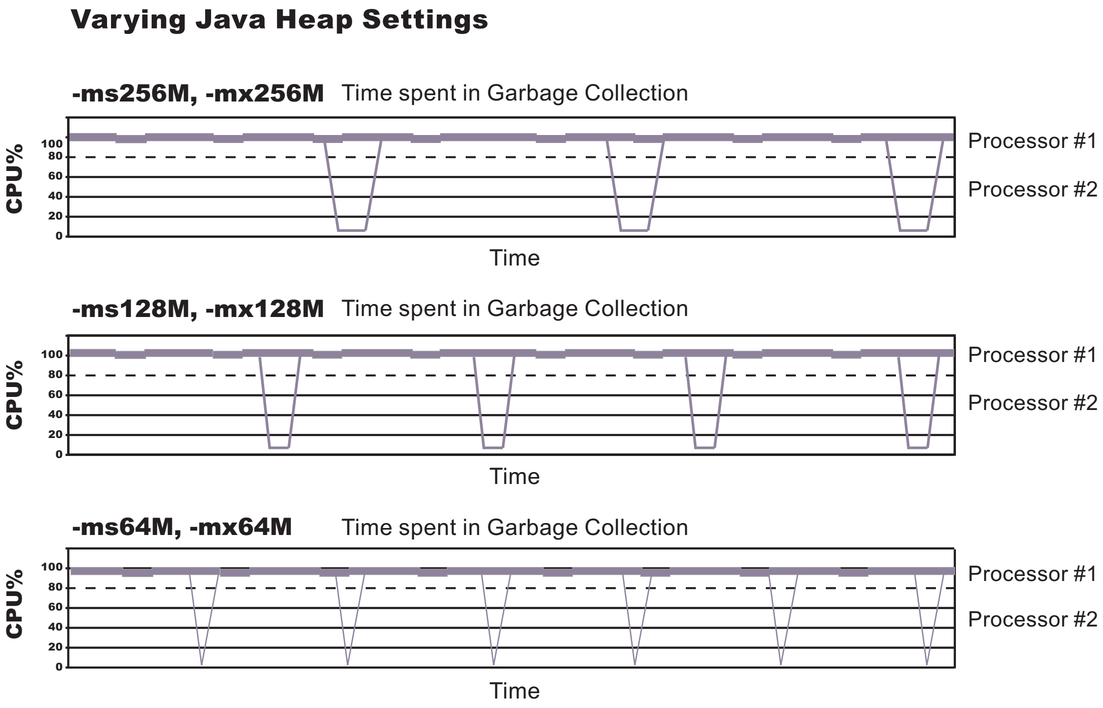
The Java heap parameters also influence the behavior of garbage collection. Increasing the heap size supports more object creation. Because a large heap takes longer to fill, the application runs longer before a garbage collection occurs. However, a larger heap also takes longer to compact and causes garbage collection to take longer. See the *Tuning guide* PDF for more information about heap settings.

Java Heap information is contained in SMF records and can be viewed dynamically using the console command DISPLAY,JVMHEAP.

For performance analysis, the initial and maximum heap sizes should be equal.

Initial heap size

When tuning a production system where the working set size of the Java application is not understood, a good starting value for the initial heap size is 25% of the maximum heap size. The JVM then tries to adapt the size of the heap to the working set size of the application.



The illustration represents three CPU profiles, each running a fixed workload with varying Java heap settings. In the middle profile, the initial and maximum heap sizes are set to 128MB. Four garbage collections occur. The total time in garbage collection is about 15% of the total run. When the heap parameters are doubled to 256MB, as in the top profile, the length of the work time increases between garbage collections. Only three garbage collections occur, but the length of each garbage collection is also increased. In the third profile, the heap size is reduced to 64MB and exhibits the opposite effect. With a smaller heap size, both the time between garbage collections and the time for each garbage collection are shorter. For all three configurations, the total time in garbage collection is approximately 15%. This example illustrates an important concept about the Java heap and its relationship to object utilization. There is always a cost for garbage collection in Java applications.

Run a series of test experiments that vary the Java heap settings. For example, run experiments with 128MB, 192MB, 256MB, and 320MB. During each experiment, monitor the total memory usage. If you expand the heap too aggressively, paging can occur. If paging occurs, reduce the size of the heap or add more memory to the system. When all the runs are finished, compare the following statistics:

- Number of garbage collection calls
- Average duration of a single garbage collection call
- Ratio between the length of a single garbage collection call and the average time between calls

If the application is not over utilizing objects and has no memory leaks, the state of steady memory utilization is reached. Garbage collection also occurs less frequently and for short duration.

If the heap free space settles at 85% or more, consider decreasing the maximum heap size values because the application server and the application are under-utilizing the memory allocated for heap.

Configuration update performance in a large cell configuration

In a large cell configuration, you might have to determine whether configuration update performance or consistency checking is more important. When configuration consistency checking is turned on, a large amount of time might be required to save a configuration change or to deploy a large number of applications. The following factors influence how much time is required:

- The more application servers or clusters there are defined in cell, the longer it takes to save a configuration change.
- The more applications there are deployed in a cell, the longer it takes to save a configuration change.

If the amount of time required to change a configuration change is unsatisfactory, you can add the config_consistency_check custom property to your JVM settings and set the value of this property to false.

1. In the administrative console, click **System administration > Deployment manager**.
2. Under Server Infrastructure, select Java and Process Management, and then click **Process Definition**.
3. Under Additional Properties, click **Java Virtual Machine > Custom Properties > New**.
4. Enter config_consistency_check in the Name field and false in the Value field.
5. Click **OK** and then **Save** to apply these changes to the master configuration.
6. Restart the server.

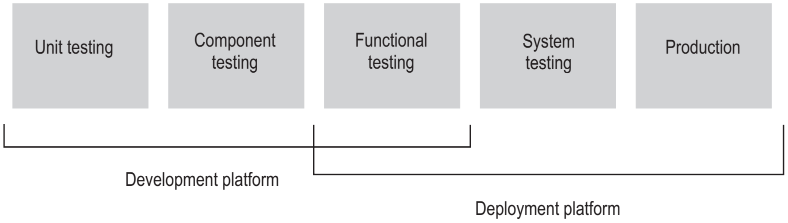
Testing and production phases

Before explaining the test and production configurations for WebSphere Application Server for z/OS, you must understand which test phase should be done on the z/OS platform and which should be done on other platforms.

Note: Sharing resources between a production workload and a test workload can expose the production workload to a set of error conditions to which it is not exposed if the production and test workloads run in different cells. If possible, you should run production and test workloads in separate cells on your system.

Before setting up your test and production configurations for WebSphere Application Server, you must understand which test phases should be done on the z/OS platform and which should be done on other platforms. The following sections explain the different phases:

- Unit test phase
- Component test phase
- Function test phase
- System test phase
- Production phase



Unit test phase

The development platform for WebSphere Application Server for z/OS is a WebSphere Application Server distributed operating system, such as Windows or Linux Intel. This development environment includes tools such as WSAD for Web content delivery. The IBM tooling solution assumes that you develop enterprise beans in WSAD and perform basic testing of the business logic in the distributed environment.

Component test phase

Component testing involves the joining together of several enterprise beans into logical components, providing them with access to data, and testing them together. While this can be done on a z/OS platform, it is recommended that you do this level of testing on a distributed platform. Performing this type of testing on a distributed platform enables a small team of developers to join the code pieces together and test the interactions. This type of testing focuses on the individual beans and their relationships to each other rather than z/OS platform functions and features.

Function test phase

Function testing involves joining the various components together, connecting them to test data in the target database, and validating the function that the application provides. Where this test is performed depends on the function and its data requirements. If the target deployment platform is z/OS, you might want to do this level of testing on the z/OS platform. In this situation you should install the applications that you are testing on one or more servers that are only used for testing.

When you install the application on a test server, define where in the JNDI directory the references to the application are stored, and then configure the test clients such that they know the location of the test application. The test clients can then drive requests against the test server to perform the functional testing. You can use remote debugging tools to diagnose problems you encounter along the way.

System test phase

Before you put an application into production on the z/OS platform, you should install the application on a WebSphere Application Server for z/OS system and simulate a real workload on that application. When setting up your system test environment, you should define an additional test server on a cell that is dedicated to the test system, and install the application onto that server. When installed, enterprise beans that are part of the application should be registered in a different subtree of the JNDI directory. This normally happens by default but it is good to verify that this registration occurs. The test clients must be configured to the version of the application that is being tested before you run your tests.

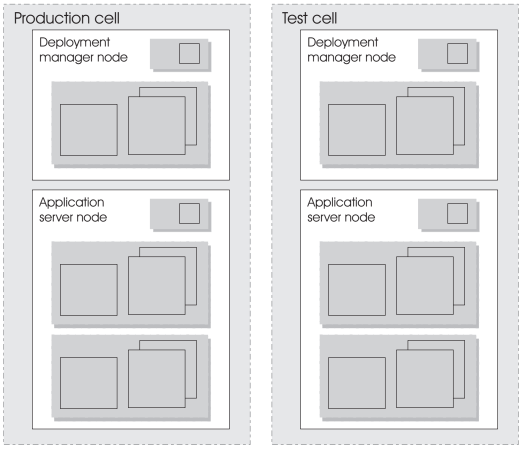
Production phase

After you are satisfied with the functional and system testing, install the application in a WebSphere Application Server for z/OS cell that is used for production . The difference between a production cell and a test cell is whether the remote debugger is allowed to be attached. Normally, it is not acceptable for a production workload to stop because a remote debugging request is sent to the cell.

Test cells and production cells

If you require complete availability of your production system, this configuration eliminates the risk of including production and test in the same cell.

As the graphic below indicates, placing test and production servers into separate cells eliminates all local sharing between test and production and provides the highest risk reduction possible.



Configuring multiple network interface support

Application servers, by default, are configured to use all of the network interfaces that are available for them to use. You can change this configuration such that an application server only uses a specific network interface. However, you cannot configure it to use a subgroup of interfaces. For example, if you have three ethernet adapters, you cannot configure an application server to use two of the three adapters.

When an application server is configured to use all network interfaces, if it opens a socket on port 9901 on a machine with two TCP/IP addresses, it opens port 9901 on both IP addresses.

When an application server is configured to use a specific network interface, it only communicates on that one network interface. For example, on a Windows operating system, if an application server opens a socket on port 7842 on an ethernet adapter with an address of 192.168.1.150, the netstat output displays 192.168.1.150.7842 in the Local Address field, indicating that port 7842 is only bound to 192.168.1.150.

If you have more than one network interface and you want to use each one separately, you must have a separate configuration profile for each interface. When network interfaces are used separately, a separate node agent is required for each network interface that has an application server running on it. Two application servers bound to two separate network interfaces on the same machine cannot be in the same node because they have different TCP/IP addresses.

Important:

- If you want a specific application server to use a single network interface, perform the following steps for that application server.
- If you want an entire node to use a single network interface, perform the following steps for your node agent and all the application servers in that node.

- If you want an entire cell to use a single network interface, perform the following steps for the deployment manager, node agent, and all the application servers in the node.
- When performing the following steps, do not specify localhost, a loop back address, such as 127.0.0.1, or an * (asterisk) for the TCP/IP addresses.

1. Update the com.ibm.CORBA.LocalHost and com.ibm.ws.orb.transport.useMultiHome Object Request Broker (ORB) custom properties.
   1. In the administrative console, navigate to the indicated page.
      - For an application server, click **Servers > Application Servers >** *server*. Then, under Container Settings, click **Container services > ORB Service** and, under Additional properties, click **Custom Properties**.
      - For a deployment manager, click **System administration > Deployment manager**, and under Additional Properties, click **ORB Service**. Then, under Additional properties, click **Custom Properties**.
      - For a node agent, click **System administration >Node agent >** *nodeagent* , and under Additional Properties, click **ORB Service**. Then, under Additional properties, click **Custom Properties**.
   2. Select the com.ibm.CORBA.LocalHost custom property and specify an IP address or hostname in the Value field. Do not set this property to either localhost or *.

      If the com.ibm.CORBA.LocalHost property is not in the list of already defined custom properties, click **New** and then enter `com.ibm.CORBA.LocalHost` in the Name field and specify an IP address or hostname in the Value field.
   3. Select the com.ibm.ws.orb.transport.useMultiHome custom property and specify `false` in the Value field. If the com.ibm.ws.orb.transport.useMultiHome property is not in the list of already defined custom properties, click **New** and then enter `com.ibm.ws.orb.transport.useMultiHome` in the Name field and specify `false` in the Value field.
2. Update the Java virtual machine (JVM) com.ibm.websphere.network.useMultiHome custom property for discovery and SOAP connections.
   1. In the administrative console, navigate to the indicated page.
      - For an application server, click **Servers > Application Servers >** *server1* **> Java and Process Management > Process Definition >** *process_type* **> Java Virtual Machine > Custom Properties**.
      - For a deployment manager, click **System Administration > Deployment manager > Java and Process Management > Process definition >** *process_type* **> Java Virtual Machine > Custom Properties**.
      - For a node agent, click **System Administration >Node agent >** *nodeagent* **> Java and Process Management > Process definition > Control > Java Virtual Machine > Custom Properties**.
   2. Select the com.ibm.websphere.network.useMultiHome custom property and specify `false` in the Value field. If the com.ibm.websphere.network.useMultiHome property is not in the list of already defined custom properties, click **New** and then enter `com.ibm.websphere.network.useMultiHome` in the Name field and specify `false` in the Value field.
3. Update the host name for TCP/IP connections.
   1. In the administrative console, navigate to the indicated page.
      - For an application server, click **Servers > Application Servers >** *server*, and then, under Communications, click **Ports**.
      - For a deployment manager, click **System administration > Deployment manager**, and then under Additional Properties, click **Ports**.
      - For a node agent, click **System administration >Node agent >** *nodeagent* , and then, under Additional Properties, click **Ports**.

   b. Update the Host field for each of the listed ports to the value specified for the com.ibm.CORBA.LocalHost ORB custom property in the first step. When you finish, none of the entries listed in the Host column should contain an * (asterisk).
4. Change the Initial State setting for each of the JMS servers to Stopped .
   a. In the administrative console, click **Servers > JMS Servers.**
   b. Click one of the listed JMS servers and change the value specified for the Initial State field to Stopped.
   c. Repeat the previous step until the Initial State setting for all of the listed JMS servers is Stopped.
5. Change the Initial State setting for each of the listener ports to Stopped .
   a. In the administrative console, click **Servers** > **Application servers** > *server*.
   b. Under Communications, click **Messaging** > **Message Listener Service** > **Listener Ports**
   c. Click one of the listed listener ports and change the value specified for the Initial State field to `Stopped`.
   d. Repeat the previous step until the Initial State setting for all of the listed listener ports is `Stopped`.
6. Save the updates and synchronize the changes with all of the node agents in the cell.
   a. In the administrative console, click **System administration** > **Save Changes to Master Repository**.
   b. Select Synchronize changes with nodes, and then click **Save**.
7. Stop and restart all the affected servers, node agents, and the deployment manager.

You have configured an installation of WebSphere Application Server to communicate on one, and only one network interface on a machine that has more than one network interface.

This example creates two nodes, each using a separate network interface, on a machine that has at least two network interfaces.
1. Use the Profile Management tool to create an application server and federate it into the desired cell.
2. Use the Profile Management tool to create an application server profile, specifying a host name that is different than the host name used for the previously created application server. Federate this application server into the desired cell.
3. Start the node agent and application server that are configured to the first network interface. Follow the preceding steps for the node agent and application server to prepare this node to communicate on the network interface you specified when you configured this application server.
4. Start the second node agent and application server. Follow the preceding steps for the node agent and application server to prepare this node to communicate only on the network interface that you specified when you configured the second application server.
5. Stop all of the node agents and application servers that you created in this example.
6. Restart all of these node agents and application servers.

You have two separate nodes running on two different network interfaces.

If you are using a standalone Java client or server to communicate with WebSphere Application Server, and you are using the WebSphere Application Server Software Development Kit (SDK), add the following properties to your Java command to enable the ORB for your application to communicate with a specific network interface.

```
-Dcom.ibm.ws.orb.transport.useMultiHome=false
-Dcom.ibm.CORBA.LocalHost=host_name
```

host_name is the TCP/IP address or *hostname* of the network interface for the ORB to use.

Important: Do not set *host_name* to localhost, a loop back address, such as 127.0.0.1, or an * (asterisk).

Tuning application servers

The WebSphere Application Server contains interrelated components that must be harmoniously tuned to support the custom needs of your end-to-end e-business application.

The following steps describe various tuning tasks that may improve your application server performance. You can choose to implement any of these application server settings. These steps can be performed in any order.

1. **Tune the object request broker.** An Object Request Broker (ORB) manages the interaction between clients and servers, using the Internet InterORB Protocol (IIOP). It supports client requests and responses received from servers in a network-distributed environment. You can use the following parameters to tune the ORB:
   - Set **Pass by reference (com.ibm.CORBA.iiop.noLocalCopies)** as described in the *Tuning guide* PDF.
   - Set the **com.ibm.CORBA.FragmentSize** as described in the *Administering applications and their environment* PDF.
2. **Tune the XML parser definitions.**
   - **Description:** Facilitates server startup by adding XML parser definitions to the jaxp.properties and xerxes.properties files in the ${*app_server_root*}/jre/lib directory. The XMLParserConfiguration value might change as new versions of Xerces are provided.
   - **How to view or set:** Insert the following lines in both files:

     ```
     javax.xml.parsers.SAXParserFactory=org.apache.xerces.jaxp.SAXParserFactoryImpl
     javax.xml.parsers.DocumentBuildFactory=org.apache.xerces.jaxp.
               DocumentBuilderFactoryImpl
     org.apache.xerces.xni.parser.XMLParserConfiguration=org.apache.xerces.parsers.
              StandardParserConfiguration
     ```
   - **Default value:** None
   - **Recommended value:** None
3. **Tune the dynamic cache service.**

   Using the dynamic cache service can improve performance. See the *Administering applications and their environment* PDF for information about using the dynamic cache service and how it can affect your application server performance.
4. **Tune the EJB container.** An Enterprise JavaBeans (EJB) container is automatically created when you create an application server. After the EJB container is deployed, you can use the following parameters to make adjustments that improve performance.
   - Set the **Cleanup interval** and the **Cache size** as described in the *Administering applications and their environment* PDF.
   - **Break CMP enterprise beans into several enterprise bean modules** while assembling EJB modules.

   See also the *Tuning guide* PDF.
5. **Tune the session management.**

   The installed default settings for session management are optimal for performance. See the *Tuning guide* PDF for more information about tuning session management.
6. **Tune the data sources and associated connection pools.** A data source is used to access data from the database; it is associated with a pool of connections to that database.
   - Review information on connection pools, contained in the *Administering applications and their environment* PDF, to understand how the number of physical connections within a connection pool can change performance.
   - Use the *Tuning guide* PDF as a reference for the data source and connection pool properties that most affect performance.
7. **Tune the URL invocation cache.**

   Each JavaServer Page is a unique URL. If you have more than 50 unique URLs that are actively being used, increase the value specified for the invocationCacheSize JVM custom property. This property controls the size of the URL invocation cache. See the *Administering applications and their environment* PDF for more information on how to change this property.

8. **Change how frequently the recovery log service attempts to compress any logstreams that application components are using.**

   The Transaction Service RLS_LOGSTREAM_COMPRESS_INTERVAL custom property can be set to a value larger then the default value if the Transaction Service is the only application component using a logstream. If none of your components are configured to use a logstream, you can set this property to 0 (zero) to disable this function.

   See the *Administering applications and their environment* PDF for more information about how to change the value specified for this property.

Web services client to Web container optimized communication

To improve performance, there is an optimized communication path between a Web services client application and a Web container that are located in the same application server process. Requests from the Web services client that are normally sent to the Web container using a network connection are delivered directly to the Web container using an optimized local path. The local path is available because the Web services client application and the Web container are running in the same process.

This direct communication eliminates the need for clients and web containers that are in the same process to communicate over the network. For example, a Web services client might be running in an application server. Instead of accessing the network to communicate with the Web container, the Web services client can communicate with the Web container using the optimized local path. This optimized local path improves the performance of the application server by allowing Web services clients and Web containers to communicate without using network transports.

In a clustered environment, there is typically an HTTP server (such as IBM HTTP server) that handles incoming client requests, distributing them to the correct application server in the cluster. The HTTP server uses information about the requested application and the defined virtual hosts to determine which application server receives the request. The Web services client also uses the defined virtual host information to determine whether the request can be served by the local Web container. You must define unique values for the host and port on each application server. You cannot define the values of host and port as wild cards denoted by the asterisk symbol (*) when you enable the optimized communication between the Web services application and the Web container. Using wild cards indicate that the local Web container can handle Web services requests for all destinations.

The optimized local communication path is disabled by default. You can enable the local communication path with the enableInProcessConnections custom property. Before configuring this custom property, make sure that you are not using wild cards for host names in your Web container end points. Set this property to **true** in the Web container to enabled the optimized local communication path. When disabled, the Web services client and the Web container communicate using network transports.

For information about how to configure the enableInProcessConnections custom property, see the *Administering applications and their environment* PDF.

When the optimized local communication path is enabled, logging of requests through the local path uses the same log attributes as the network channel chain for the Web container. To use a different log file for in process requests than the log file for network requests, use a custom property on the HTTP Inbound Channel in the transport chain. Use the inProcessLogFilenamePrefix custom property to specify a string that is added to the beginning of the network log file name to create a file name that is unique. Requests through the local process path are logged to this specified file. For example, if the log filename is ../httpaccess.log for a network chain, and the inProcesslLogFilenamePrefix custom property is set to “local” on the HTTP channel in that transport chain, the local log file name for requests to the host associated with that chain is /localhttpaccess.log.

Application servers: Resources for learning

Use the following links to find relevant supplemental information about configuring application servers. 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 instructions and examples
- Programming specifications
- Administration

Programming instructions and examples

- WebSphere Application Server education

Programming specifications

- The Java^{TM} Virtual Machine Specification, Second Edition
- Sun's technology forum for the Java^{TM} Virtual Machine Specification

Administration

- Listing of all IBM WebSphere Application Server Redbooks

Chapter 8. Balancing workloads with clusters

You should use server clusters and cluster members to monitor and manage the workloads of application servers.

Consider your options for configuring application servers. See “Managing application servers” on page 253 for more information.

To assist you in understanding how to configure and use clusters for workload management, consider this scenario. Client requests are distributed among the cluster members on a single machine. A *client* refers to any servlet, Java application, or other program or component that connects the end user and the application server that is being accessed.)

In more complex workload management scenarios, you can distribute cluster members within the same sysplex.

1. Decide which application server you want to cluster.
2. Decide whether you want to replicate data. Replication is a service that transfers data, objects, or events among application servers. See “Replicating data across application servers in a cluster” on page 416 for more information. You can create a replication domain when creating a cluster.
3. Deploy the application onto the application server.
4. After configuring the application server and the application components exactly as you want them to be, create a cluster. The original server instance becomes a cluster member that is administered through the cluster. See “Creating clusters” on page 397 for more information.
5. You can create one or more cluster members of the cluster.
6. Start all of the application servers by starting the cluster. Workload management automatically begins when you start the cluster members of the application server.
7. Once you have the cluster running, you can perform the following tasks:
   - Stop the cluster.
   - Upgrade applications on clusters. See the *Administering applications and their environment* PDF for more information.
   - Detect and handle problems with server clusters and their workloads.

Clusters and workload management

Clusters are sets of servers that are managed together and participate in workload management. The servers that are members of a cluster can be on different host machines, as opposed to the servers that are part of the same node and must be located on the same host machine. A cell can have no clusters, one cluster, or multiple clusters.

Servers that belong to a cluster are *members* of that cluster set and must all have identical application components deployed on them. Other than the applications configured to run on them, cluster members do not have to share any other configuration data. One cluster member might be running on a huge multi-processor enterprise server system, while another member of that same cluster might be running on a smaller system. The server configuration settings for each of these two cluster members are very different, except in the area of application components assigned to them. In that area of configuration, they are identical. This allows client work to be distributed across all the members of a cluster instead of all workload being handled by a single application server.

When you create a cluster, you make copies of an existing application server template. The template is most likely an application server that you have previously configured. You are offered the option of making that server a member of the cluster. However, it is recommended that you keep the server available only as a template, because the only way to remove a cluster member is to delete the application server. When you delete a cluster, you also delete any application servers that were members of that cluster. There is no way to preserve any member of a cluster. Keeping the original template intact allows you to reuse the template if you need to rebuild the configuration.

A *vertical cluster* has cluster members on the same node, or physical machine. A *horizontal cluster* has cluster members on multiple nodes across many machines in a cell. You can configure either type of cluster, or have a combination of vertical and horizontal clusters.

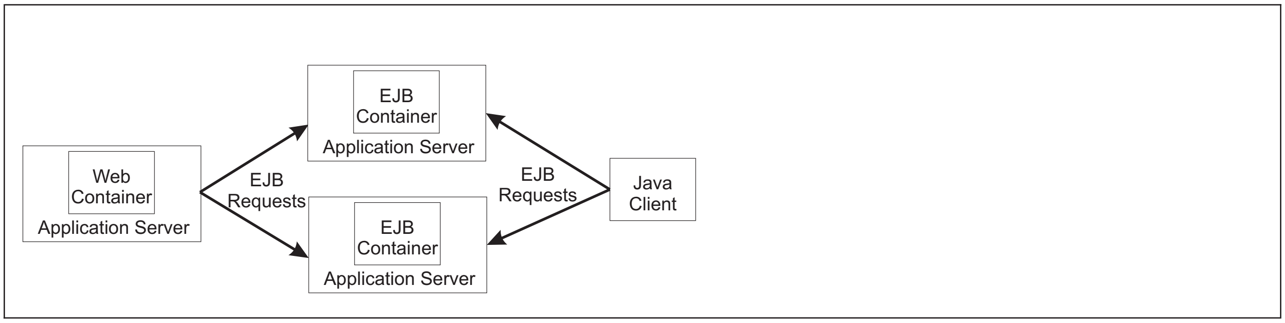
Assign a weight to a cluster member based on its approximate, proportional ability to do work. The weight value specified for a specific member is only meaningful in the context of the weights you specify for the other members within a cluster. The weight values do not indicate absolute capability. If a cluster member is unavailable, the Web server plug-in temporarily routes requests around that cluster member.

For example, if you have a cluster that consists of two members, assigning weights of 1 and 2 causes the first member to get approximately 1/3 of the workload and the second member to get approximately 2/3 of the workload. However, if you add a third member to the cluster, and assign the new member a weight of 1, approximately 1/4 of the workload now goes to the first member, approximately 1/2 of the workload goes to the second member, and approximately 1/4 of the workload goes to the third member. If the first cluster member becomes unavailable, the second member gets approximately 2/3 of the workload and third member gets approximately 1/3 of the workload.

The weight values only approximate your load balance objectives. There are other application dependencies, such as thread concurrency, local setting preferences, affinity, and resource availability that are also factors in determining where a specific request is sent. Therefore, do not use the exact pattern of requests to determine the weight assignment for specific cluster members.

See “Cluster member settings” on page 412 for information on how to set the weight for a cluster member.

Workload management for EJB containers can be performed by configuring the Web container and EJB containers on separate application servers. Multiple application servers can be clustered with the EJB containers, enabling the distribution of enterprise bean requests between EJB containers on different application servers.



In this configuration, EJB client requests are routed to available EJB containers in a round robin fashion based on assigned server weights. The EJB clients can be servlets operating within a Web container, stand-alone Java programs using RMI/IIOP, or other EJBs.

The server weighted round robin routing policy ensures a balanced routing distribution based on the set of server weights that have been assigned to the members of a cluster. For example, if all servers in the cluster have the same weight, the expected distribution for the cluster is that all servers receive the same number of requests. If the weights for the servers are not equal, the distribution mechanism sends more requests to the higher weight value servers than the lower weight value servers. The policy ensures the desired distribution, based on the weights assigned to the cluster members.

You can set up workload management to balance the tasks between different clusters.

You can choose to have requests sent to the node on which the client resides as the preferred routing. In this case, only cluster members on that node are chosen (using the round robin weight method). Cluster members on remote nodes are chosen only if a local server is not available

WebSphere Application Server can respond to increased use of an enterprise application by automatically replicating the application to additional cluster members as needed. This lets you deploy an application on a cluster instead of on a single node, without considering workload.

Multiple servers that can service the same client request form the basis for failover support. If a server fails while processing a client request, the failed request can be rerouted to any of the remaining cluster members. In fact, several servers could fail, and as long as at least one cluster member is running, client requests can continue to be serviced. For further information backing up failed processes, see “Replicating data across application servers in a cluster” on page 416.

Clusters and node groups

Node groups bound clusters. All cluster members of a given cluster must be members of the same node group.

Any application you install to a cluster must be able to execute on any application server that is a member of that cluster. For the application you deploy to run successfully, all the members of the cluster must be located on nodes that meet the requirements for that application.

In a cell that has many different server configurations, it might be difficult to determine which nodes have the capabilities to host your application. A *node group* can be used to define groups of nodes that have enough in common to host members of a given cluster. All cluster members in a cluster must be in the same node group.

All nodes are members of at least one node group. When you create a cluster, the first application server you add to the cluster defines the node group that bounds the cluster. All other cluster members you add to the cluster can only be on nodes that are members of this same node group. When you create a new cluster member in the administrative console, you are allowed to create the application server on a node that is a member of the node group for that cluster only.

Nodes can be members of multiple node groups. If the first cluster member you add to a cluster has multiple node groups defined, the system automatically chooses the node group that bounds the cluster. You can change the node group by modifying the cluster settings. Use the “Server cluster settings” on page 406 page to change the node group.

Workload management (WLM) for z/OS

Workload management optimizes the distribution of incoming work requests to the application servers, enterprise beans, servlets, and other objects that can most effectively process the requests. Workload management also provides failover when servers are not available, improving application availability.

For details on workload management, see *z/OS MVS Planning: Workload Management*, which is available on the z/OS Internet Library Web site. You might also find *z/OS MVS Programming: Workload Management Services* helpful.

When you are using workload management on z/OS, you can define workload management policies for your application servers. To get started, you do not need to define special classification rules and work qualifiers, but you might want to define them for your production system.

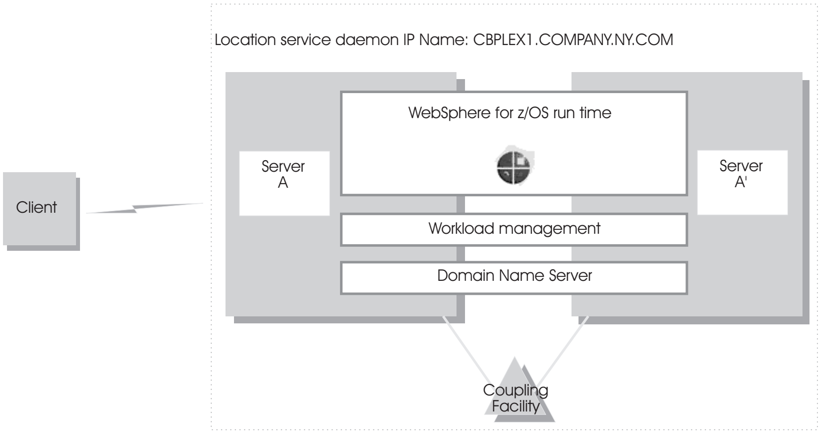
Workload management provides the following benefits to WebSphere Application Server applications:
- It balances server workloads, allowing processing tasks to be distributed according to the capacities of the different machines in the sysplex.

- It provides failover capability by redirecting client requests if one or more servers is unable to process them. This improves the availability of applications and administrative services.
- It enables systems to be scaled up to serve a higher client load than provided by the basic configuration. With clustering, additional instances of servers, servlets, and other objects can easily be added to the configuration.
- It enables servers to be transparently maintained and upgraded while applications remain available for users.
- It centralizes the administration of servers and other objects.

In the WebSphere Application Server environment, you implement workload management by using clusters, transports, and replication domains.

Connection optimization

Characteristics of a configuration in which the Domain Name Server cooperates with workload management (WLM) to route client requests throughout a cell are:


Location service daemon IP Name: CBPLEX1.COMPANY.NY.COM
WebSphere for z/OS run time
Server A
Server A'
Client
Workload management
Domain Name Server
Coupling Facility

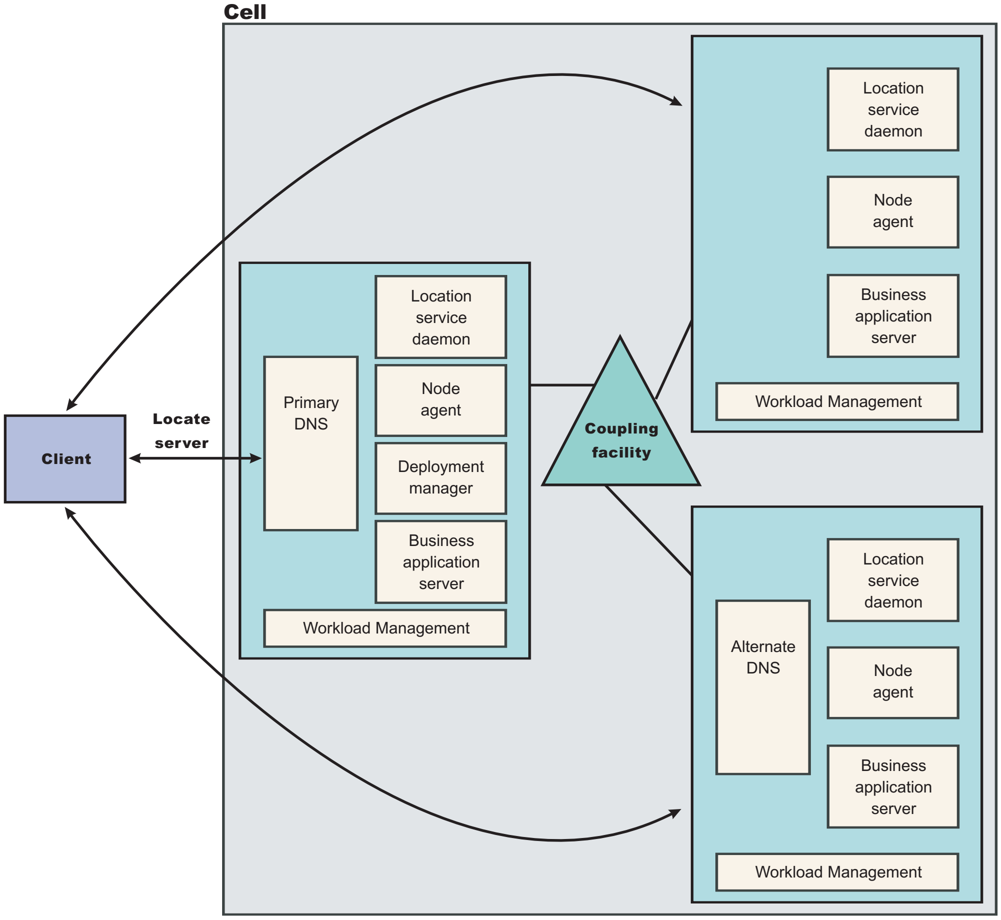
- The domain name server (DNS) is replicated by setting up a secondary DNS on more than one system in the cell.
- The client must know the host name and port of the name server to connect to WebSphere Application Server for z/OS.
- Each system in the cell has the same location service daemon IP name. Workload management and the domain name server determine the actual system that receives client requests. The client sees the cell as a single system, though its requests might be balanced across systems in the cell.
- As part of workload balancing and maximizing performance goals, workload management also routes work requests to systems in the cell. This function is possible because WebSphere Application Server for z/OS cooperates with workload management. Because the system references that a client sees are indirect, even requests from that same client might be answered by differing systems in the cell.
- The implication for clients is that they should not cache IP addresses unless they can recover from failed connections. That is, if a connection fails, a client should be able to reissue a request, but, because the IP address is an indirect address, a reissue of the request can be answered by another system in the cell.

For more information, see “Workload management (WLM) for z/OS” on page 373. For additional details on setting up servers for connection optimization, see *z/OS Communications Server: IP Configuration Reference*.

Sysplex routing of work requests

WebSphere Application Server for z/OS uses the domain name server (DNS) to route work requests within a cell. You can use the DNS, instead of a sysplex distributor to distribute workload and balance requests for the same hostname across multiple IP addresses (one per daemon).

The DNS accepts a generic hostname from the client and maps the name to a specific system. The DNS works with workload management (WLM) to select the best available system . Workload management analyzes the current state of the cell and considers a number of factors, such as CPU, memory, and I/O utilization, when it determines the best system to handle new work. The DNS then routes the client request to that system. This use of workload management and the DNS is optional. However, using workload management and the DNS eliminates a single point of failure.



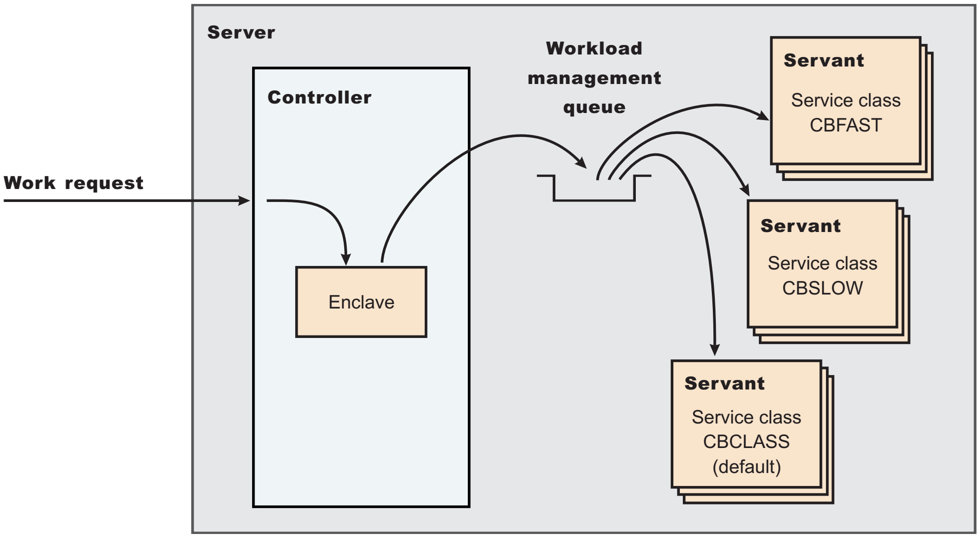
Each system in a cell has the WebSphere Application Server for z/OS runtime. All the systems contain a location service daemon, node agent, and business application servers. One system acts as the deployment manager for the cell. The client uses the CORBA General Inter-ORB Protocol (GIOP) to make requests of WebSphere Application Server for z/OS. The location service daemon acts as a location service agent. It accepts locate requests with object keys in the requests. The location service daemon extracts the server cluster name from the object key, then gives the server name to workload management. Workload management chooses the optimal server in the cell to handle the request. The location service daemon merges specific Interoperable Object Reference (IOR) information that is related to the chosen server with object key information stored in the original IOR. The result of this merging is a direct IOR that gets returned to the client. The client ORB uses this returned reference to establish the IOR connection to the server holding the object of interest.

The transport mechanism that WebSphere Application Server for z/OS uses depends on whether the client is local or remote. If the client is remote, that is, the client is not running on the same z/OS system as WebSphere Application Server for z/OS, the transport is TCP/IP. If the client is local, the transport is through a program call. Local transport is faster because it does not require a physical trip over the network, eliminates data transforms, simplifies the marshalling of requests, and uses optimized RACF facilities for security rather than having to invoke Kerberos or SSL.

Address space management for work requests

WebSphere Application Server for z/OS propagates the performance context of work requests by using workload management (WLM) enclaves. Each transaction has its own enclave and is managed according to its service class.

The controller of a server, which workload management views as a queue manager, uses the enclave associated with a client request to manage the priority of the work. If the work has a high priority, workload management can direct the work to a high-priority servant in the server. If the work has a low priority, workload management can direct the work to a low-priority servant. The effect is to partition the work according to priority within the same server.



Enclaves can originate in several ways:
- WebSphere Application Server for z/OS uses its own set of rules to create an enclave for a client request from the network.
- Some subsystems, such as the IBM HTTP Sever, create enclaves and pass them to the application server, which, in turn, passes the enclaves on.
- WebSphere Application Server for z/OS treats batch jobs as if they were remote clients.

To communicate the performance context to workload management, you must classify the workloads in your system according to the following work qualifiers.

Table 8. WLM work qualifiers and corresponding WebSphere Application Server for z/OS entities

Work qualifier abbreviation	Work qualifier	Corresponding WebSphere Application Server for z/OS entity
CN	Collection name	Cluster name
UI	User ID	User ID under which work is running

For more information about classification rules and workload qualifiers, see *z/OS MVS Planning: Workload Management* and “Classifying z/OS workload” on page 381.

In addition to client workloads, you must consider the performance of the WebSphere Application Server for z/OS run-time servers and your business application servers. In general, server controllers act as work routers, so they must have high priority. Because workload management starts and stops servants dynamically, servants also need high priority to be initialized quickly. After the servants are initialized, they run work according to the priority of the client enclave, so the servant priority that you assign has no significance after initialization.

In summary, use the following table to set the performance goals for each class:

Table 9. Workload management rules

If you are classifying...	... assign it to:	Explanation
Location service daemon	SYSSTC or a high velocity, high importance STC	The system treats it as a started task, and it must route work requests quickly.
WebSphere Application Server controller	SYSSTC or a high velocity, high importance STC	A controller must route work quickly, but you must balance the priority of your business application server with other work in the system.
WebSphere Application Server servant	A lower velocity and importance STC than the controller	If too high, you spend too much time in Java garbage collection, and you might consume more system resources than you want. If too low, JSP compiles and Java garbage collection could delay processing in your application. Startup of additional servant address spaces might also be delayed.
WebSphere Application Server application environment	Use the CB classification rules, the percentage response time goal, for example 80% of transactions complete in .25 seconds.	
Client applications	Assuming a long-running application, a velocity goal should be used that is relative to other work on the system.	

Example of classification rules

In this example, all work for BBOC001, except for work running under the user ID DBOOZ, gets classified as CBFAST. Work for DBOOZ gets classified as CBSLOW. All other work, such as work coming from clients outside the cell and including the work for WebSphere Application Server for z/OS runtime servers, gets classified as CBCLASS.

Purpose

Let us assume you have three workload management service classes defined for WebSphere Application Server for z/OS (subsystem type CB):

1. CBFAST-designed for transactions requiring fast response times.
2. CBSLOW-designed for long-running applications that do not require fast response times.
3. CBCLASS-designed for remaining work requests.

You design a client workload called BBOC001 that requires fast response times. Also, you want to give work that runs under your manager’s user ID (DBOOZ) slower response times. Finally, all remaining work requests should run under the default service class, CBCLASS.

Table 10. Classification rules example

Type column	Name column	Service column	Goal
CN	BBOC001	CBFAST	90% complete in 2 seconds

Table 10. Classification rules example (continued)

Type column	Name column	Service column	Goal
UI	DBOOZ	CBSLOW	Velocity 50, importance = 3
(default)	(blank)	CBCLASS	Discretionary

You could set the following performance goals through IWMARIN0:

1. Issue IWMARIN0 and choose option 4:

```
  File  Utilities  Notes  Options  Help
--------------------------------------------------------------------------
  Functionality LEVEL003   Definition Menu  WLM Appl LEVEL004    Command ===>
______________________________________________________________

Definition data set  . . : 'CB.MYCB.WLM'
Definition name  . . . . . CB390      (Required)
Description  . . . . . . . WLM Setup for WebSphere for z/OS
Select one of the following options. . . . . 4__
1.  Policies
2.  Workloads
3.  Resource Groups
4.  Service Classes
5.  Classification Groups
6.  Classification Rules
7.  Report Classes
8.  Service Coefficients/Options
9.  Application Environments
10.  Scheduling Environments
```

2. Create a service class called CBFAST and specify that it be 90% complete in 2 seconds.

   Note: The example assumes you have defined a workload called ONLINE.

```
      Service-Class  Notes  Options  Help
--------------------------------------------------------------------------
Create a Service Class
Row 1 to 2 of 2   Command ===> ____________________________________________________
Service Class Name . . . . . . CBFAST    (Required)
Description  . . . . . . . . . Quick CB transactions
Workload Name  . . . . . . . . ONLINE    (name or ?)
Base Resource Group  . . . . . ________  (name or ?)
Specify BASE GOAL information.  Action Codes: I=Insert new period,
E=Edit period, D=Delete period.
---Period---  ---------------------Goal---------------------
Action  #  Duration   Imp.  Description
__                                                                       __
1             1    90% complete within 00:00:02.000
 ****************************** Bottom of data ********************************

.-----------------------------------------------------------------------.
| Press EXIT to save your changes or CANCEL to discard them. (IWMAM970) |
'-----------------------------------------------------------------------'
```

3. Save the service class. You see the following:

```
      Service-Class  View  Notes  Options  Help
--------------------------------------------------------------------------
Service Class Selection List
Row 1 to 14 of 21   Command ===> ______________________________________________________
Action Codes: 1=Create, 2=Copy, 3=Modify, 4=Browse, 5=Print, 6=Delete,
/=Menu Bar
Action  Class
Description
Workload
__    CBFAST
```

```
 Quick CB Transactions
ONLINE
******************************* Bottom of data ********************************
```

4. Repeat these steps for the CBSLOW service class.
5. Create classification rules using the new service class. Choose option 6 on the main panel:

```
     File  Utilities  Notes  Options  Help
--------------------------------------------------------------------------
Functionality LEVEL003        Definition Menu        WLM Appl LEVEL004
Command ===> ______________________________________________________________
Definition data set  . . : 'CB.MYCB.WLM'
Definition name  . . . . . CB390      (Required)
Description  . . . . . . . WLM Setup for WebSphere for z/OS
Select one of the following options. . . . . 6__
1.  Policies
2.  Workloads
3.  Resource Groups
4.  Service Classes
5.  Classification Groups
6.  Classification Rules
7.  Report Classes
8.  Service Coefficients/Options
9.  Application Environments
10.  Scheduling Environments
```

6. Create a set of rules for your service classes:

```
     Subsystem-Type  Xref  Notes  Options  Help
--------------------------------------------------------------------------
Create Rules for the Subsystem Type       Row 1 to 2 of 2
Command ===> ___________________________________________   SCROLL ===> PAGE
Subsystem Type . . . . . . . . CB    (Required)
Description  . . . . . . . . .  WebSphere  classification
Fold qualifier names?  . . . . Y  (Y or N)
Action codes:  A=After    C=Copy        M=Move     I=Insert rule
B=Before   D=Delete row   R=Repeat   IS=Insert Sub-rule
 -------Qualifier-------------
-------Class--------
Action    Type       Name     Start
Service     Report
DEFAULTS: CBCLAS      ________
____  1  CN
BBOC001    ___
CBFAST      ________
____  1  UI
DBOOZ     ___
CBSLOW      ________
***************************** BOTTOM OF DATA ******************************
```

Enabling multiple servants on z/OS

Use this task to enable multiple servants on your WebSphere Application Server for z/OS system. Enabling multiple servants improves the performance of WebSphere Application Server for z/OS.

Review the limitations of enabling multiple servants.

Workload Management (WLM), enables you to manage the performance and number of servants in WebSphere Application Server for z/OS. WLM manages the response time and throughput of transactions according to their assigned service class, the associated performance objectives, and the availability of system resources. To meet these goals, WLM sometimes needs to control or override the number of servants that are active. With this task, you can enable WLM to start additional servants to meet performance goals.

1. In the administrative console, click **Servers > Application servers >** *server_name*.
2. Under Server Infrastructure, click **Java and Process Management > Server Instance**.

3. Select the Multiple instances enabled field.
4. Click **Apply** to finish the Server Instance changes.
5. Click **OK** and then click **Save** to save your configuration changes.

Workload Management (WLM) can start additional servant regions to meet performance goals, based on the importance of its work compared to other work in the system, the availability of system resources that are needed to satisfy those objectives, and a determination by WLM of whether starting more address spaces might help achieve the objectives. It is also important to make the goals reasonable.

Multiple servant regions

With Workload Management (WLM), you can manage the performance and number of servants in WebSphere Application Server for z/OS. WLM manages the response time and throughput of WebSphere transactions according to their assigned service class, the associated performance objectives, and the availability of system resources. To meet these goals, WLM sometimes needs to control or override the number of servants that are active.

WebSphere Application Server applications are deployed within a WebSphere Application Server generic *server*. One or more server instances must be defined on one or more systems within the WebSphere Application Server *node*. Each server instance consists of a *controller* and one or more *servants*. The controllers are started by MVS as *started tasks*, and servants are started by WLM, as they are needed.

If you have WLM dynamic application environments enabled on your system (z/OS Release 2 plus APAR OW54622 or later), WLM honors the specifications for the number of servant regions. If you are using static application environments (specified through the WLM ISPF panels), then you must also enable multiple servant regions by indicating **No limit** in the WLM ISPF panels.

Note: Be aware of the following regarding multiple servants:

- Some applications, such as the administrative console and the deployment manager, have serialization requirements that only work in a single Java virtual machine (JVM). These cannot be run in a server with multiple servants. Do not enable multiple instances for any of these servers.
- If you specify a maximum number of instances, WLM is restricted from starting more than this number of servant regions for this server instance.
- **The maximum number of servants should be at least as large as the number of different service classes that might be used by transactions that are run in the server**. Remember to account for the *default CB-type service class* and enclaves that may originate outside WebSphere Application Server servers and are classified by other classification rules such as the IBM HTTP Server (IHS).

Controlling the number of servants

You can control the minimum or maximum number of servants for a server using the administrative console. The minimum value is useful for starting up a basic number of servants before your work arrives. This can reduce delays while waiting for Workload Manager (WLM) to start up additional servants.

The maximum value is useful for *capping* the number of address spaces that are started by WLM for each *server instance*, if you determine that excessive servant regions are contributing to service degradation.

1. Start the administrative console.
2. Click **Servers > Application servers >** ***server_name***.
3. Under Server Infrastructure, click **Java and Process Management --> Server Instance**
4. Type a value into the **Minimum Number of Instances** and **Maximum Number of Instances** fields, or leave these fields blank to allow WLM to determine the numbers.
5. Click **Apply** to finish the Server Instance changes.
6. Click **OK**.

Classifying z/OS workload

You can use a common workload classification document to classify inbound HTTP, IIOP, and message-driven bean (MDB) work requests for the z/OS workload manager.

You should use workload management on a z/OS system. See "Workload management (WLM) for z/OS" on page 373 for more information.

A workload classification document file is an XML file in which you classify incoming HTTP, IIOP, and message-driven bean (MDB) work requests and assign them to a transaction class (TCLASS). The TCLASS value, if it is assigned, is passed to the MVS Workload Manager. WLM uses the TCLASS value to classify the inbound work requests and to assign a service class or a report service class to each request.

The common workload classification document is the method you should use to classify work requests in a z/OS environment. Support for other WebSphere Application Server mechanisms for classifying work in a z/OS environment is deprecated and you should no longer use those mechanisms.

If you want to classify work for message-driven beans deployed against JCA 1.5 resources with the default messaging provider, or you want to classify mediation work for use with service integration buses, you need to define a Classification element that uses SibClassification elements. You must also perform z/OS Workload Manager actions that are required to use the TCLASS value "SIBUS". If you replace any listener port with a JMS activation specification for use by MDB applications with the version 6 default messaging provider, you should replace any related InboundClassification type=″mdb″ classifications with SibClassifications type=″jmsra″ classifications.

1. Develop the workload classification document. Use the information in the article,"Workload classification file" on page 386. See "Sample z/OS workload classification document" on page 384 for a sample of the workload classification document and the DTD.
2. If you create the document on a z/OS system in codepage IBM-1047, the normal codepage for files that exist in the HFS, you must convert the file to ASCII before you use the file. Use one of the following options to convert a working document into a document that can be used by the server:
   - native2ascii

     This is a utility in the Java SDK that can convert a file from the native codepage to the ASCII codepage. For example, if you are working on an XML document called x5sr02.classification.ebcdic.xml and you want to create a document called x5sr02.classification.xml, use the following command:

     ```
     /u/userid -> native2ascii \
     x5sr02.classification.ebcdic.xml > x5sr02.classification.xml
     ```

     The command line is split with the backslash (\) character to the next line for publication purposes.
   - iconv

     This is a z/OS utility that can convert files from one designated codepage to a different designated codepage. For example, if you are working on an XML document called x5sr02.classification.ebcdic.xml and you want to create a document called x5sr02.classification.xml, use the following command:

     ```
     /u/userid -> iconv -f IBM-1047 -t UTF-8 \
     x5sr02.classification.ebcdic.xml >x5sr02.classification.xml
     ```

     The command line is split with the backslash (\) character to the next line for publication purposes.
   - Create the document on your workstation and then FTP the file to the correct location on the z/OS system in binary format. By using this option, you can also create the Classification.dtd file in the same directory as the workload classification document. Then, you can perform an XML validity

check on the document before installing it on a server. Use any type of validating parser, for example, you can use WebSphere Application Developer workbench to construct and validate the workload classification document.

3. Specify the location of the workload classification document in the administrative console. Use the wlm_classification_file variable to specify the XML file that contains the classification information. In the administrative console, click **Environment > Manage WebSphere variables > New**. You can set the variable at cell, node, or server instance level. If you specify the variable at the cell or node level, the information must be accessible and applicable to all the servers that inherit the specification from the node or cell.
4. You must perform z/OS Workload Manager actions that are required to use the TCLASS values. Each TCLASS must be assigned a service class, report service class, or both to the enclave under which the work runs. The CB classification rules must be updated.

   If you want to classify work for message-driven beans deployed against JCA 1.5 resources with the default messaging provider, or you want to classify mediation work for use with service integration buses, you need to perform z/OS Workload Manager actions that are required to use the TCLASS value “SIBUS”.

   Transaction classes are used as sub-rules in establishing service classes and transaction. The TCLASS values are not used as level one rules. If you decide to use TCLASS as a level one rule rather than a sub-rule, you must be careful in ordering the rules. The first level one rule that applies to the work is used, so more specific rules should be first, followed by the broad rules. For example, consider the following two examples of CB classification rules:

```
Subsystem-Type Xref Notes Options Help
--------------------------------------------------------------------------
Modify Rules for the Subsystem Type Row 1 to 17 of 17
Command ===> _____________________________________________ SCROLL ===> CSR
Subsystem Type . : CB Fold qualifier names? Y (Y or N)
Description . . . CB Class'n w/WLM Trans. CLASSes
Action codes: A=After C=Copy M=Move I=Insert rule
B=Before D=Delete row R=Repeat IS=Insert Sub-rule
More ===>
       --------Qualifier--------              -------Class--------
Action   Type   Name   Start                      Service Report
                  DEFAULTS:  CBCLASS RWASDEF
____ 1  CN  P5SR01*  1                        CBCLASS RTP5CLUS
____ 1  TC      A0      ___                      CBHUTCH RP5A0
____ 1  TC      A1      ___                      CBHUTCH RP5A1
____ 1  TC      A1B     ___                      CBHUTCH RP5A1B
____ 1  CN    WSIVP2*   ___                      CBSLOW  RWSIVP
____ 1  CN    T%SERV*   1                        CBFAST  RTSMIGT
____ 1  CN      B4*     ___                      CBFAST ________
```

   In the preceding example, the TCLASS assignments that are made for enclaves running in the server P5SR01x are never used by the workload manager. When the following rule is run, no further searching of the classification table is done:

```
____ 1  CN    P5SR01*  1            CBCLASS
```

   The TCLASS assignments are not used. All of the enclaves that run in the P5SR01x servers are assigned to the CBCLASS service class and the RTP5CLUS report service class.

```
Subsystem-Type Xref Notes Options Help
--------------------------------------------------------------------------
Modify Rules for the Subsystem Type Row 1 to 17 of 17
Command ===> _____________________________________________ SCROLL ===> CSR
Subsystem Type . : CB Fold qualifier names? Y (Y or N)
Description . . . CB Class'n w/WLM Trans. CLASSes
Action codes: A=After C=Copy M=Move I=Insert rule
B=Before D=Delete row R=Repeat IS=Insert Sub-rule
More ===>
         --------Qualifier--------           -------Class--------
Action   Type    Name    Start               Service       Report
                                  DEFAULTS: CBCLASS       RWASDEF
```

```
____ 1   TC      A0     ___                CBHUTCH      RP5A0
____ 1   TC       A1       ___                CBHUTCH      RP5A1
____ 1   TC      A1B      ___                CBHUTCH      RP5A1B
____ 1  CN     P5SR01*     1                 CBCLASS      RTP5CLUS
____ 1   CN     WSIVP2*    ___                CBSLOW       RWSIVP
____ 1   CN     T%SERV*     1                 CBFAST       RTSMIGT
____ 1   CN       B4*      ___                CBFAST       ________
```

In the preceding example, if a TCLASS value of A0, A1, or A1B are provided in the classification, they are used regardless of which server is running the work. In this case, the server name is used only if these three TCLASS values are not present.

5. Restart the application server to implement any changes to the file. If the workload classification document is not a well formed, valid XML document it is ignored by the application server and the following message is displayed:

   ```
   BBOJ0085E PROBLEMS ENCOUNTERED PARSING WLM CLASSIFICATION XML FILE (0)
   ```

6. Use the DISPLAY WORK operator command to display classification information. Use this command to determine if your classification scheme is classifying the work as you intended. Issue the following command to display the IIOP and HTTP classification information:

   ```
   MODIFY|F <servername>, DISPLAY,WORK,CLINFO
   ```

   Issue this command against each application server.

   The following example shows a possible result of issuing the new operator command:

   ```
   F X5SR02A,DISPLAY,WORK,CLINFO

   BBOO0281I CLASSIFICATION COUNTERS FOR IIOP WORK
   BBOO0282I CHECKED 14, MATCHED 14, USED 0, COST 0, DESC: IIOP Default
   BBOO0282I CHECKED 14, MATCHED 3, USED 0, COST 0, DESC: sample
   BBOO0282I CHECKED 3, MATCHED 1, USED 1, COST 3, DESC: a1a
   BBOO0282I CHECKED 2, MATCHED 1, USED 1, COST 4, DESC: a1b
   BBOO0282I CHECKED 1, MATCHED 1, USED 1, COST 5, DESC: a1c
   BBOO0282I CHECKED 11, MATCHED 11, USED 0, COST 0, DESC: other
   BBOO0282I CHECKED 11, MATCHED 1, USED 1, COST 4, DESC: a
   BBOO0282I CHECKED 10, MATCHED 1, USED 1, COST 5, DESC: b
   BBOO0282I CHECKED 9, MATCHED 1, USED 1, COST 6, DESC: c
   BBOO0282I CHECKED 8, MATCHED 2, USED 2, COST 7, DESC: d
   BBOO0282I CHECKED 6, MATCHED 1, USED 1, COST 8, DESC: e
   BBOO0282I CHECKED 5, MATCHED 4, USED 4, COST 9, DESC: f
   BBOO0282I CHECKED 1, MATCHED 1, USED 1, COST 10, DESC: g
   BBOO0283I FOR IIOP WORK: TOTAL CLASSIFIED 14, WEIGHTED TOTAL COST 95
   BBOO0281I CLASSIFICATION COUNTERS FOR HTTP WORK
   BBOO0282I CHECKED 0, MATCHED 0, USED 0, COST 0, DESC: HTTP Default
   BBOO0282I CHECKED 0, MATCHED 0, USED 0, COST 0, DESC: n
   BBOO0282I CHECKED 0, MATCHED 0, USED 0, COST 0, DESC: o
   BBOO0282I CHECKED 0, MATCHED 0, USED 0, COST 0, DESC: q
   BBOO0282I CHECKED 0, MATCHED 0, USED 0, COST 0, DESC: r
   BBOO0282I CHECKED 0, MATCHED 0, USED 0, COST 0, DESC: s
   BBOO0282I CHECKED 0, MATCHED 0, USED 0, COST 0, DESC: p
   BBOO0282I CHECKED 0, MATCHED 0, USED 0, COST 0, DESC: t
   BBOO0283I FOR HTTP WORK: TOTAL CLASSIFIED 0, WEIGHTED
   BBOO0188I END OF OUTPUT FOR COMMAND DISPLAY,WORK,CLINFO
   ```

   An explanation of the command output follows:

   BBOO0281I CLASSIFICATION COUNTERS FOR *type* **WORK**

   The header message for messages that display the usage of the workload classification rules. The value of *type* can be HTTP or IIOP. You cannot display inbound MDB classifications.

   BBOO0282I CHECKED *n1*, **MATCHED** *n2*, **USED** *n3*, **COST** *n4*, **DESC:** *text*

   This message displays information about a particular rule in the workload classification. This message displays the following information:

   - *n1* - The number of times the rule has been examined.

- *n2* - The number of this times that this rule has been matched by the request.
- *n3* - The number of times that this rule has actually been used.
- *n4* - The cost of using the rule, or the number of compares that are required to determine if this is the correct rule to use.
- *text* - The descriptive text from the classification rule so that you can tell which classification rule is being displayed.

BBOO0283I FOR *type* **WORK: TOTAL CLASSIFIED** *n1***, WEIGHTED TOTAL COST** *n2*

This message shows the summary information for the HTTP or IIOP work classification. This message displays the following information:

- *type* - The type of work that is being displayed. The value must be IIOP or HTTP.
- *n1* - The number of requests that were classified using the classification rules.
- *n2* - The weighted total cost, calculated by taking the number of times that each rule was used multiplied by the cost, or number of rule compares that were done, of using the rule and adding those up across all of the rules.

The total cost *n2* divided by the total number of requests classified *n1* equals the cost of using the table. The closer that the value is to one, the lower the cost of using the defined rules. A value of 1 indicates that there is just the default classification, so no requests match it.

7. Repeat these steps until you achieve your optimal workload distribution and costs.

By completing this task, you classified inbound requests by using a workload classification document.

See Chapter 8, “Balancing workloads with clusters,” on page 371 for more information about configuring workload management.

Sample z/OS workload classification document

Sample

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Classification SYSTEM "Classification.dtd" >
<Classification schema_version="1.0">
<!--
        IIOP Classification Rules
-->
   <InboundClassification type="iiop"
                          schema_version="1.0"
                          default_transaction_class="A0">
     <iiop_classification_info transaction_class="A1"
                               application_name="IIOPStatelessSampleApp"
                               module_name"StatelessSample.jar"
                               component_name="Sample20"
                               description="Sample20 EJB Classification">
       <iiop_classification_info  transaction_class=""
                                  method_name="echo"
                                  description="No TCLASS for echo()" />
       <iiop_classification_info  transaction_class="A1B"
                                 method_name="ping"
                                 description="Ping method" />
     </iiop_classification_info>
     <iiop_classification_info application_name="*"
                               module_name="*"
                               component_name="*"
                               transaction_class="A2"
                               description="TCLASS the rest to A2">
        <iiop_classification_info  transaction_class="A2A"
                                   method_name="resetFilter"
                                   description="Sp1 case resetFilter()" />
     </iiop_classification_info>
  </InboundClassification>
```

```
<!--
     HTTP Classification Rules
-->
   <InboundClassification  type="http"
                           schema_version="1.0"
                           default_transaction_class="M">
     <http_classification_info  transaction_class="N"
                                host="yourhost.yourcompany.com"
                                description="Virtual Host yourhost">
       <http_classification_info transaction_class="O"
                                 port="9080"
                                 description="Def yourhost HTTP reqs">
         <http_classification_info  transaction_class="Q"
                                    uri="/gcs/admin"
                                    description = "Gcs" />
         <http_classification_info  transaction_class="S"
                                    uri="/gcs/admin/1*"
                                    description="GCS login" />
     </http_classification_info>
     <http_classification_info  transaction_class="P"
                                port="9081"
                               description=" Def yourhost HTTPS reqs "/>
       <http_classification_info  transaction_class=""
                                  uri="/gcss/mgr/*"
                                  description="GCSS Mgr" />
     </http_classification_info>
   </http_classification_info>
  </InboundClassification>

<!--
     MDB Classification Rules
-->
  <InboundClassification  type="mdb"
                           schema_version="1.0"
                           default_transaction_class="qrs">
     <endpoint  type="messagelistenerport"
               name="IVPListenerPort"
               defaultclassification="MDBX"
               description="ABC">
        <classificationentry  selector="Location&apos;East&apos;"
                              classification="MDB1"
                              description="DEF"/>
        <classificationentry  selector="Location&lt;&gt;&apos;East&apos;"
                              classification="MDB2"
                              description="XYZ" />
     </endpoint>
     <endpoint  type="messagelistenerport"
                name="SimpleMDBListenerPort"
                defaultclassification="MDBX"
                description="GHI" />
   </InboundClassification>

    <SibClassification type="jmsra" schema_version="1.0"
        default_transaction_class="a">
        <sib_classification_info transaction_class="b"
            selector="user.Location=&apos;East&apos;" bus="magic"
            destination="nowhere" description="n" />
        <sib_classification_info transaction_class="c"
            selector="user.Location=&apos;West&apos;" bus="omni" description="n" />
    </SibClassification>
    <SibClassification type="destinationmediation" schema_version="1.0"
        default_transaction_class="b">
        <sib_classification_info transaction_class="e"
            selector="user.Location=&apos;East&apos;" destination="themoon"
            discriminator="sides/dark" description="n" />
        <sib_classification_info transaction_class="f"
```

```
            selector="user.Location=&apos;West&apos;" description="n" />
    </SibClassification>

<!--
   Workload Classification Document for P5SR01x servers
   Change History
   -------------------------------------------------------
   Activity                    Date                Author
   Created                    01-28-2005          IPL

--->

</Classification>
```

DTD for the workload classification XML document

```
From SERV1/ws/code/messaging.impl.ws390/src/Classification.dtd
SERV1/ws/code/messaging.impl.ws390/src/Classification.dtd                    WAS601.SERV1      WAS.messaging      1.2
<?xml version='1.0' encoding="UTF-8"?>
<!ELEMENT Classification (InboundClassification|SibClassification)+>
<!ATTLIST Classification schema_version CDATA #REQUIRED>
<!ELEMENT InboundClassification ((iiop_classification_info*|http_classification_info*|endpoint*))>
<!ATTLIST InboundClassification type (iiop|mdb|http) #REQUIRED>
<!ATTLIST InboundClassification default_transaction_class CDATA #REQUIRED>
<!ATTLIST InboundClassification schema_version CDATA #REQUIRED>
<!ELEMENT iiop_classification_info (iiop_classification_info*)>
<!ATTLIST iiop_classification_info activity_workload_classification CDATA #IMPLIED>
<!ATTLIST iiop_classification_info application_name CDATA #IMPLIED>
<!ATTLIST iiop_classification_info component_name CDATA #IMPLIED>
<!ATTLIST iiop_classification_info description CDATA #IMPLIED>
<!ATTLIST iiop_classification_info method_name CDATA #IMPLIED>
<!ATTLIST iiop_classification_info module_name CDATA #IMPLIED>
<!ATTLIST iiop_classification_info transaction_class CDATA #REQUIRED>
<!ELEMENT endpoint (classificationentry*)>
<!ATTLIST endpoint defaultclassification CDATA #REQUIRED>
<!ATTLIST endpoint name CDATA #REQUIRED>
<!ATTLIST endpoint type (messagelistenerport) #REQUIRED>
<!ATTLIST endpoint description CDATA #IMPLIED>
<!ELEMENT classificationentry EMPTY>
<!ATTLIST classificationentry classification CDATA #REQUIRED>
<!ATTLIST classificationentry selector CDATA #REQUIRED>
<!ATTLIST classificationentry description CDATA #IMPLIED>
<!ELEMENT http_classification_info (http_classification_info*)>
<!ATTLIST http_classification_info host CDATA #IMPLIED>
<!ATTLIST http_classification_info port CDATA #IMPLIED>
<!ATTLIST http_classification_info uri CDATA #IMPLIED>
<!ATTLIST http_classification_info description CDATA #IMPLIED>
<!ATTLIST http_classification_info transaction_class CDATA #REQUIRED>
<!ELEMENT SibClassification (sib_classification_info+)>
<!ATTLIST SibClassification type (jmsra|destinationmediation) #REQUIRED>
<!ATTLIST SibClassification default_transaction_class CDATA #REQUIRED>
<!ATTLIST SibClassification schema_version CDATA #REQUIRED>
<!ELEMENT sib_classification_info EMPTY>
<!ATTLIST sib_classification_info transaction_class CDATA #REQUIRED>
<!ATTLIST sib_classification_info selector CDATA #IMPLIED>
<!ATTLIST sib_classification_info bus CDATA #IMPLIED>
<!ATTLIST sib_classification_info destination CDATA #IMPLIED>
<!ATTLIST sib_classification_info discriminator CDATA #IMPLIED>
<!ATTLIST sib_classification_info description CDATA #IMPLIED>
```

Workload classification file

The workload classification document is a common XML file that classifies inbound HTTP, IIOP, message-driven bean (MDB), and mediation work for the z/OS workload manager.

Usage notes

You must perform the task “Classifying z/OS workload” on page 381 when you use the workload classification document. See “Sample z/OS workload classification document” on page 384 for an example of a workload classification document.

Required elements

<?xml version=″1.0″ encoding=″UTF-8″>
: Indicates that the workload classification document must be saved in ASCII to be processed by the application server. This statement is required.

<!DOCTYPE Classification SYSTEM ″Classifications″>
: Gives the XML parser with the name of the DTD document provided by WebSphere Application Server for z/OS that validates the workload classification document. The workload classification document that you write must follow the rules that are described in this DTD. You must add this statement to the workload classification document.

Classification

<Classification schema_version=″1.0″>

Indicates the root of the workload classification document. Every workload classification document must begin and end with this element. The schema_version attribute is required. The only supported schema_version is 1.0. The Classification element contains one or more InboundClassification elements. For inbound service integration work, the Classification element can also contain up to two SibClassification elements.

InboundClassification

<InboundClassification type=″iiop | http | mdb″ schema_version=″1.0″ default_transaction_class=″value″>

Use the following rules when using the InboundClassification element:

- The **type** attribute is required. The value must be iiop, http, or mdb. Only one occurrence of an InboundClassification element can occur in the document for each type. There can be up to three InboundClassification elements in a document. The types do not have to be specified in a certain order in your classification document.
- The **schema_version** attribute is required. The value must be set to 1.0.
- The **default_transaction_class** attribute must be specified, and defines the default transaction class for work flows of the specified type. The string value must be a valid WLM transaction class, a null string (such as ″″) or a string that contains eight or fewer blanks (such as ″ ″).
- The InboundClassification elements cannot be nested. Each InboundClassification element must end before the next InboundClassification element or SibClassification element can begin.

SibClassification

<SibClassification type=″jmsra | destinationmediation″ schema_version=″1.0″ default_transaction_class=″value″>

Use the following rules when using the SibClassification element:

- The **type** attribute is required. The value must be jmsra or destinationmediation. There can be at most one SibClassification element in the document for each type. The types do not have to be specified in a certain order in your classification document.
- The **schema_version** attribute is required. The value must be set to 1.0.
- The **default_transaction_class** attribute must be specified, and defines the default transaction class for work flows of the specified type. The string value must be a valid WLM transaction class, a null string (such as ″″) or a string that contains eight or fewer blanks (such as ″ ″).

- The SibClassification elements cannot be nested. Each SibClassification element must end before the next InboundClassification element or SibClassification element can begin.

The rules and XML statements for classifying different types of work are very similar, but there is slightly different syntax for each type. For more information about the syntax for each type of work, see the following sections:

InboundClassification

- "IIOP Classification"
- "HTTP classification" on page 390
- "MDB classification" on page 391

SibClassification

- "JMS RA classification" on page 392
- "Mediation classification" on page 393

IIOP Classification

The InboundClassification element with the attribute type="iiop" defines the section of the document that is applicable to IIOP classification. An example of this element follows:

```
<InboundClassification  type="iiop" schema_version="1.0"
                        default_transaction_class="value1">
```

You can classify IIOP work based on the following Java 2 Platform, Enterprise Edition (J2EE) application artifacts:

- Application name

  The name of the application that contains the enterprise beans. It is the display name of the application, which might not be the name of the .ear file that contains all of the artifacts.
- Module name

  The name of the EJB .jar file that contains one or more enterprise beans. There can be multiple EJB .jar files in an .ear file.
- Component name

  The name of the EJB that is contained in a module (or EJB .jar file). There can be one or more enterprise beans contained in a .jar file.
- Method name

  The name of a remote method on an EJB.

Classify IIOP work in various applications at any of these levels by using the iiop_classification_info element.

iiop_classification_info

```
<iiop_classification_info  transaction_class="value1"
                           [application_name="value2"]
                           [module_name="value3"]
                           [component_name="value4"]
                           [method_name="value5"]
                           [description="value6"] >
```

With the iiop_classification_info element, you can build filters based on the application, module, component, and method names to assign TCLASS values to inbound requests. Use the following rules when using the iiop_classification_info element:

- The transaction_class attribute must be specified. The string value must be a valid WLM transaction class, a null string (such as ″″) or a string that contains eight or fewer blanks (such as ″ ″). By specifying a null or blank string, you can override a default TCLASS setting, or a

TCLASS setting that was assigned by a higher level filter. Specifying a null or blank string means that you do not have a TCLASS value for the request.

- The attributes application_name, module_name, component_name, and method_name can be used as you need them. These attributes act as selectors or filters that either assign a transaction class or allow a nested iiop_classification_info element to assign the transaction class. You can specify the values of these attributes in the following ways:
  - The exact name of the application, module, component, or method.
  - A wild carded value. You can use an asterisk (*) to match strings. The string MAY* matches any string that starts with MAY.
  - Any value. To specify a match to any value, use the asterisk (*) symbol.

  Use any or all of these attributes to make a classification filter. Only use the granularity that is required. For example, if there is only one application on the application server, the classification rules do not need to specify the application_name attribute.
- The iiop_classification_info elements can be nested in a hierarchical manner. By nesting the elements, you can create classification filters that are based on the attribute values. The following filter classifies requests on the EJB1 and EJB2 enterprise beans in the MyAPP1 application:

```
<iiop_classification_info transaction_class="FAST"
                          application_name="MyAPP1"
                          component_name="EJB1" />
<iiop_classification_info transaction_class="SLOW"
                          application_name="MyAPP1"
                          component_name="EJB2" />
```

  The following filter also classifies requests on EJB1 and EJB2 in the MyAPP1 application, but also classifies requests on any other EJB in the application:

```
<iiop_classification_info transaction_class="MEDIUM"
                          application_name="MyAPP1">
     <iiop_classification_info transaction_class="FAST"
                               component_name="EJB1" />
     <iiop_classification_info transaction_class="SLOW"
                               component_name="EJB2" />
</iiop_classification_info>
```

- If you specify an attribute value that conflicts with the parent element's attribute value, the lower level filter is negated. An example of a child value that conflicts with the parent element's attribute value follows:

```
<iiop_classification_info transaction_class="FAST"
                          application_name="MyAPP1">
     <iiop_classification_info transaction_class="SLOW"
                               application_name="MyAPP2" />
</iiop_classification_info>
```

  In this example, EJB Requests in MyAPP2 would never be assigned to transaction class ″SLOW″ because the higher level filter only allows IIOP requests for application_name=″MyAPP1″ to be passed through to the lower level filter.
- The first filter at a specific level that matches the attributes of the request is used, not the best or most restrictive filter. Therefore, the order that you specify filters is important.

```
<iiop_classification_info transaction_class="FAST"
                          application_name="MyAPP" />
     <iiop_classification_info transaction_class="SLOW"
                               component_name="*" />
     <iiop_classification_info transaction_class="MEDIUM"
                               component_name="MySSB" />
</iiop_classification_info>
```

  In the preceding example, all the IIOP requests that are processed by enterprise beans in the MyAPP application are assigned a TCLASS value of SLOW. This is true for any requests to the

MySSB enterprise as well. Even though MySSB is assigned a transaction class, the filter is not applied because the first filter was applied and was assigned a TCLASS value of SLOW. The remaining list of filters at the same level is ignored.

- The description field is optional. However, you should use a description on all iiop_classification_info elements. The description string prints as part of the operator command support so you can identify the classification rules that are being used. Keep your descriptions reasonably short because they are displayed in the MVS console.

HTTP classification

The InboundClassification element with the attribute type=″http″ defines the section of the document that is applicable to HTTP classification. An example of this element follows:

```
<InboundClassification   type="http"
                        schema_version="1.0"
                        default_transaction_class="value1">
```

HTTP work can be classified based on the following J2EE artifacts:

- Virtual host name

  Specifies the host name in the HTTP header to which the inbound request is being sent.
- Port number

  Specifies the port on which the HTTP catcher is listening.
- URI (Uniform Resource Identifier)

  The string that identifies the Web application.

You can classify HTTP work in various applications at any of these levels by using the http_classification_info element.

```
<http_classification_info transaction_class="value1"
                          [host="value2"]
                          [port="value3"]
                          [uri="value4"]
                          [description="value5"] >
```

With the http_classification_info element, you can build filters based on the host, port, and URI to assign TCLASS values to inbound requests. Use the following rules when you use the http_classification_info element:

- The transaction_class attribute must be specified. The string value must be a valid WLM transaction class, a null string (such as ″″) or a string that contains eight or fewer blanks (such as ″ ″). By specifying a null or blank string, you can override a default TCLASS setting, or a TCLASS setting that was assigned by a higher level filter. Specifying a null or blank string means that you do not have a TCLASS value for the request.
- The attributes host, port, and uri can be used as you need them. These attributes act as selectors or filters that either assign a transaction class or allow a nested http_classification_info element to assign the transaction class. You can specify the values of these attributes in the following ways:
  - The exact name of the host, port, or uri.
  - A wild carded value. You can use an asterisk (*) to match strings. The string MAY* matches any string that starts with MAY.
  - Any value. To specify a match to any value, use the asterisk (*) symbol.

  Use any or all of these attributes to make a classification filter. Only use the granularity that is required. For example, if there is only one application on the application server, the classification rules do not need to specify the uri attribute.
- You can nest the http_classification_info elements in a hierarchical manner. You can construct filters based on attribute names. Consider the two following filters:

```
<http_classification_info transaction_class="FAST"
                          host="MyVHost1.com"
                          uri="/MyWebApp1/*" />
<http_classification_info transaction_class="SLOW"
                          host="MyVHost2.com"
                          uri="/MyWebApp2/*" />

<http_classification_info transaction_class="MEDIUM"
                          host="MyVHost1.com">
     <http_classification_info transaction_class="FAST"
                               uri="/MyWebApp1/*" />
     <http_classification_info transaction_class="SLOW"
                               uri="/MyWebApp2/*" />
</http_classification_info>
```

Both filters classify requests to Web applications that are identified by context roots /MyWebApp1 and /MyWebApp2 in the application server that is hosting web applications for virtual host MyVHost1.com. However, the second filter also classifies requests on any other context root in the application server.

- Specifying an attribute name that is different from the parent element's attribute value effectively negates the lower level filter. For example:

```
<http_classification_info transaction_class="FAST"
                          uri="/MyWebApp1/*">
     <http_classification_info transaction_class="SLOW"
                               uri="/MyWebApp2">
     </http_classification_info>
</http_classification_info>
```

  This example would never result in Web applications with a context root of /MyWebApp2 being assigned to the transaction class SLOW. The high level filter only allows HTTP requests with a context root of /MyWebApp1/* to be passed to a lower level filter.

- The first filter that is at a specific level is used, not the best or most restrictive filter. Therefore, the order of the filters at each level is important. For example:

```
<http_classification_info transaction_class="FAST"
                          host="MyVHost.com" />
     <http_classification_info transaction_class="SLOW"
                               uri="*" />
     <http_classification_info transaction_class="MEDIUM"
                               uri="/MyWebAppX/*" />
</http_classification_info>
```

  In this example, HTTP requests processed by the application server by the virtual host ″MyVHost.com″ are assigned a TCLASS value of SLOW. Even requests to the Web application with context root /MyWebAppX are assigned a TCLASS value of SLOW because the filter was not applied. The first filter that matches is used for the TCLASS assignment, and the remainder of the filters at the same level is ignored.

- The description field is optional, however, you should use it on all the http_classification_info elements. The description is displayed when you monitor the transaction classes in the MVS console.

MDB classification

The InboundClassification element with the attribute type=″mdb″ defines the section of the document that applies to work for EJB 2.0 message-driven beans (MDBs) deployed with listener ports. An example of this element follows:

```
<InboundClassification  type="mdb"
                        schema_version="1.0"
                        default_transaction_class="qrs">
```

Each InboundClassification element can contain one or more endpoint elements with a messagelistenerport type defined. Define one endpoint element for each listener port that is defined in the server that you want to associate transaction classes with the message-driven bean. An example of the endpoint element follows:

```
<endpoint  type="messagelistenerport"
           name="IPVListenerPort"
           defaultclassification="MDBX"
           description="ABC">
```

Use the following rules when defining your endpoint elements:

- The type attribute must always equal messagelistenerport.
- The name attribute corresponds to the listener for your endpoint element. The value of the name attribute must be the name of the listener port that is specified in the administration console for the server.
- The defaultclassification element is the default transaction class that is associated with the message-driven beans. The value of this attribute overrides the default transaction classification value.
- The description field is optional, however, you should use it on all the endpoint elements. The description is displayed when you monitor the transaction classes in the MVS console.

Each endpoint element can have zero, one, or more classificationentry elements. An example of a classification entry element follows:

```
<classificationentry  selector="&apos;East&apos;"
                      classification="MDB1"
                      description="DEF" />
```

Use the selector attribute of the classificationentry element to assign a transaction class to a message-driven bean that has a selector clause in its deployment descriptor. Use the following rules when defining your classificationentry elements:

- The value of the selector attribute must match exactly to the selector clause in the MDB deployment descriptor.
- The value of the selector attribute must have the correct syntax for an XML document. You must replace the < and > symbols with the entity references &lt; and &gt;, respectively. Similarly, if you use an apostrophe or quotation mark, use the &apos; and &quot; entity references.

JMS RA classification

The SibClassification element with the attribute type=″jmsra″ defines the section of the document that applies to work for message-driven beans (MDBs) deployed against JCA 1.5-compliant resources for use with the JCA resource adapter (RA) of the default messaging provider. An example of this element follows:

```
<SibClassification  type="jmsra"
                        schema_version="1.0"
                        default_transaction_class="a">
```

Each SibClassification element can contain one or more sib_classification_info elements. An example of a classification entry element follows:

```
<sib_classification_info  selector="&apos;East&apos;"
                      transaction_class="sibb"
                      selector="user.Location=&apos;East&apos;"
                      bus="bigrred"
                      destination="abusqueue"
                      description="Some words" />
```

selector

Use the selector attribute of the sib_classification_info element to assign a transaction class to a message-driven bean that has a selector clause in its deployment descriptor. Use the following rules when defining your sib_classification_info elements:

- The value of the selector attribute is an SQL expression that selects a message according to the values of the message properties. The syntax is that of a message selector in the JMS 1.1 specification, but it can operate on SIMessage messages (more than JMS messages). The syntax can select on system properties (including JMS headers, JMSX properties, and JMS_IBM_properties) and user properties (which must be prefixed by ".user" - for example, for the user property "Location", the selector would specify "user.Location" as shown in the preceding example). For more information, see Working with the message properties.
- The value of the selector attribute must have the correct syntax for an XML document. You must replace the < and > symbols with the entity references &lt; and &gt;, respectively. Similarly, if you use an apostrophe or quotation mark, use the &apos; and &quot; entity references.

bus The name of the service integration bus on which the target destination is assigned. The classification applies to the bus named by this property, or to any bus if you do not specify this property. The destinations to which the classification applies depend on your use of the destination property.

destination
The name of the target bus destination to which the message has been delivered. This is the name of a queue or topic space. The classification applies to the destination named by this property, or any destination if you do not specify this property. The service integration buses to which the classification applies depend on your use of the bus property.

discriminator
The property applies only when the destination property names a topic space. This discriminator value is then an XPath expression that selects one or more topics within the topic space.

description
Although the description field is optional, you should use it on all the sib_classification_info elements. The description is displayed when you monitor the transaction classes in the MVS console.

Each sib_classification_info element can contain one or more of these properties as needed to classify the work for a message. A sib_classification_info element cannot contain more than one instance of each property.

If a message matches several sib_classification_info elements, the element that appears first is used. For example, consider the following specifications:

```
<sib_classification_info bus="MyBus" transaction_class="a" />
<sib_classification_info destination="MyDest" transaction_class="b" />
```

A message that arrives at destination MyDest from the service integration bus MyBus is assigned the classification "a". A message that arrives at MyDest from another bus is assigned the classification "b".

If a message does not match any sib_classification_info element in an enclosing SibClassification element, the message is assigned the default classification from the SibClassification element.

If a message does not match any sib_classification_info element in any SibClassification element, or if no SibClassification elements are defined, all work receives a built-in default classification with the value "SIBUS". You must perform z/OS Workload Manager actions that are required to use the TCLASS value "SIBUS", as described in "Classifying z/OS workload" on page 381.

Mediation classification

The SibClassification element with the attribute type=″destinationmediation″ defines the section of the document that applies to work for mediations assigned to destinations on a service integration bus. An example of this element follows:

```
    <SibClassification type="destinationmediation"
                       schema_version="1.0"
                       default_transaction_class="b">
```

Each SibClassification element can contain one or more sib_classification_info elements. An example of a classification entry element follows:

```
<sib_classification_info
                    transaction_class="e"
                    selector="user.Location=&apos;East&apos;"
                    destination="themoon"
                    discriminator="sides/dark"
                    description="n" />
```

selector
Use the selector attribute of the sib_classification_info element to assign a transaction class to a mediation that has a selector clause in its deployment descriptor. Use the following rules when defining your sib_classification_info elements:

- The value of the selector attribute is an SQL expression that selects a message according to the values of the message properties. The syntax is that of a message selector in the JMS 1.1 specification, but it can operate on SIMessage messages (more than JMS messages). The syntax can select on system properties (including JMS headers, JMSX properties, and JMS_IBM_properties) and user properties (which must be prefixed by ".user" - for example, for the user property "Location", the selector would specify "user.Location" as shown in the preceding example).
- The value of the selector attribute must have the correct syntax for an XML document. You must replace the < and > symbols with the entity references &lt; and &gt;, respectively. Similarly, if you use an apostrophe or quotation mark, use the &apos; and &quot; entity references.

bus
The name of the service integration bus on which the target destination is assigned. The classification applies to the bus named by this property, or to any bus if you do not specify this property. The destinations to which the classification applies depend on your use of the destination property.

destination
The name of the target bus destination to which the message has been delivered. This is the name of a queue or topic space. The classification applies to the destination named by this property, or any destination if you do not specify this property. The service integration buses to which the classification applies depend on your use of the bus property.

discriminator
The property applies only when the destination property names a topic space. This discriminator value is then an XPath expression that selects one or more topics within the topic space.

description
Although the description field is optional, you should use it on all the sib_classification_info elements. The description is displayed when you monitor the transaction classes in the MVS console.

Each sib_classification_info element can contain one or more of these properties as needed to classify the work for a message. A sib_classification_info element cannot contain more than one instance of each property.

If a message matches several sib_classification_info elements, the element that appears first is used. For example, consider the following specifications:

```
<sib_classification_info transaction_class="e" destination="themoon" description="n" />
<sib_classification_info transaction_class="f" description="n" />
```

A message that arrives at the mediated destination themoon is assigned the classification "e". A message that arrives at another mediated destination is assigned the classification "f".

If a message does not match any sib_classification_info element in an enclosing SibClassification element, the message is assigned the default classification from the SibClassification element.

If a message does not match any sib_classification_info element in *any* SibClassification element, or if *no* SibClassification elements are defined, all work receives a built-in default classification with the value “SIBUS”. You must perform z/OS Workload Manager actions that are required to use the TCLASS value “SIBUS”, as described in “Classifying z/OS workload” on page 381.

Classifying WebSphere transaction workload for WLM

This topic describes how to use transaction classes to classify client workload for workload management. The workload consists of different WebSphere transactions targeted to separate servant regions, each with goals defined by appropriate service classes. Each transaction is dispatched in its own WLM enclave in a servant region process, and is managed according to the goals of its service class.

Important: Transaction class mapping file support is deprecated. You should use a workload classification document instead of a transaction class mapping file to classify work requests in a z/OS environment.

This topic describes steps to classify transaction workload as a way of managing the workload service objectives. You also need to define the service objectives (goals) for the service classes used. In addition, you must define the service objectives of the WebSphere Application Server for z/OS servers and your business application servers.

For more information about defining service objectives (goals) for each service class, see the *z/OS MVS Planning: Workload Management* book, SA22-7602, for example at http://publibz.boulder.ibm.com/epubs/pdf/iea2w131.pdf, or the z/OS WLM Web page at http://www.ibm.com/servers/eserver/zseries/zos/wlm/.

You can classify your WebSphere work using the WLM CB-type classification criteria:

- Server name (CN)
- Server instance name (SI)
- User ID assigned to the transaction (UI)
- Transaction class (TC)

Note: To get started, you do not need to define special classification rules and work qualifiers, but you may want to do this for your production system.

To classify work using server and userid criteria, you use a combination of the WLM Workload Classification rules in the WLM ISPF dialog panels. For more information about defining WLM Classification rules, see Workload management (WLM) and its related article that includes an example of classification rules.

To classify work using transaction classes, you define and use transaction class mappings, as described in this task. The following steps are used to classify work using transaction classes:

1. Define transaction class mappings based on the HTTP virtual host name, port number, and URI (Universal Resource Identifier - encoded address for any resource on the Web) provided with each work HTTP or HTTPS request.
   a. Create a Transaction Class mapping file (as a simple text file). For example: `/wasconfig/t5was/MyTrMapFile.txt`
   b. Edit the Transaction Class mapping file to define each transaction class mapping that you want to use. Define each mapping on a separate line, using the following syntax:

      ```
      TransClassMap host:port uritemplate tclass
      ```

      Note: In the host or port fields, you can use wildcard characters only for the entire field as shown in the following example.

For more information about this syntax, see Transaction class mapping file entries. For example:

```
TransClassMap wsc4.washington.ibm.com:9080  /MyIVT/index.*    TCLMYIVT
TransClassMap wsc4.washington.ibm.com:9080  /MyIVT/ivtejb     TCLMYEJB
TransClassMap wsc4.washington.ibm.com:*     /SuperSnoop*      TCLSNOOP
TransClassMap wsc4.washington.ibm.com:*     /ssb/*            TCLSSB
TransClassMap *:*                           /admin*           TCLADMIN
```

2. Specify the Transaction Class mapping file on the administrative properties for each server that is to handle work classified by transaction class. To specify the Transaction Class mapping file for a server, use the administrative console to complete the following steps:
   a. In the navigation pane, click **Servers > Application Servers**.
   b. In the content pane, select the server instance, *server_name*.
   c. In the Additional Properties list in the contents pane, select **Web Container**.
   d. In the Additional Properties list for the Web container, select **Advanced Settings**.
   e. In the **Transaction Class Mapping** field, type the name of the Transaction Class mapping file that you edited in an earlier step. For example: `/wasconfig/t5was/MyTrMapFile.txt`
   f. If you want to use a transaction class to classify outbound data that is delivered in response to HTTP and HTTPS requests, select the TCLASS option in the **Network QoS** field. If you specify TCLASS, WebSphere Application Server for z/OS uses the transaction class value that was used to classify the inbound request to the z/OS Workload Manager.

The following table shows classification rules for CB-type work in which the default service class is WSMED and has a reporting class of RWSDEFLT. Work run in the WSPROD WebSphere server is classified as WSMED with a reporting class of RWSPROD, unless it has a transaction class of TCLASS1, TCLASS2, or TCLASS2 assigned through the transaction class mapping file below.

```
Qualifier    Qualifier Start       Service  Report
# type       name      position    Class    Class
- ---------  --------  --------    -------- --------
                          Default: WSMED    RWSDEFLT
1 CN         WSPROD    1           WSMED    RWSPROD
2 . TC       . TCLASS1             WSFAST   RWSPRD1
2 . TC       . TCLASS2             WSMED    RWSPRD2
2 . TC       . TCLASS5             WSSLOW   RWSPRD5
1 CN         WSTEST    1           WSSLOW   RTSTEST
2 . UI       . USER1               WSMED    RTSTSTU2
2 . TC       . TCLASS5             WSSLOW   RTSTST5
```

The following table shows how work can be assigned a transaction class based on its host name, port number, or URI. For example, a web request of http://ibm.com:80/Webap1/myservlet handled by the WSPROD server would be assigned a transaction class of TCLASS1, a service class of WSFAST, and a reporting class of RWSPRD1 by the classification rules shown above.

```
TransClassMap www.ibm.com:80 /Webap1/myservlet TCLASS1
TransClassMap www.ibm.com:* /Webap1/myservlet TCLASS2
TransClassMap *:443 * TCLASS3
TransClassMap *:* /Webap1/myservlet TCLASS4
TransClassMap www.ibm.com:* /Webap5/* TCLASS5
TransClassMap * * TCLASS6
```

Transaction class mapping file entries

Important: Transaction class mapping file support is deprecated. You should use a workload classification document instead of a transaction class mapping file to classify work requests in a z/OS environment.

Following is the syntax for entries in a transaction class mapping file:

```
TransClassMap host:port uritemplate tclass
```

where:

host Is the value compared against the hostname of the HOST: header of the request. This value can be a wildcard '*'.

Note: A value of '*' for the host:port value is acceptable and is equivalent to '*:*'.

port Is the value compared against the port of the request. This value can be a wildcard '*'.

uritemplate
Is the value compared against the URI of the request. Any query string will not be used in the comparison. This value can be a wildcard '*', or end in a wildcard.

tclass Is the Workload Manager Transaction Class name that will be used in the creation of the enclave.

Examples:

```
TransClassMap www.ibm.com:80 /webap1/myservlet TCLASS1

TransClassMap www.ibm.com:* /webap1/myservlet TCLASS2

TransClassMap *:443 * TCLASS3

TransClassMap *:* /webap1/myservlet TCLASS4

TransClassMap www.ibm.com:* /webap2/* TCLASS5

TransClassMap * /myservlet TCLASS6

TransClassMap * * TCLASS6
```

Controller and Servant WLM classifications

You should classify the WebSphere Application servant regions to a high STC importance service class so that they can be initialized quickly when WLM determines they are needed. The service class chosen also determines the WLM goal when Java Garbage Collection (GC) is running, which can be CPU intensive. You do not want to set a servant higher in the service class hierarchy than more important work such as production WebSphere, CICS, or IMS transaction servers.

WebSphere application controller regions do some processing to receive work into the system, manage the HTTP transport handler, classify the work and do other housekeeping tasks. Therefore, controller regions should also be classified in SYSSTC or a high importance and velocity goal.

Here is a simple example of the WLM Classification Rules for STC-type work that covers the controller and servant regions started tasks:

```
          --------Qualifier--------           -------Class--------
Action    Type     Name     Start             Service     Report
                                    DEFAULTS: OPS_DEF     ________
_____  1  TN      %%DMN    ___                OPS_HIGH    RWSDMN _____  1  TN      T5SRV*   ___                OPS_MED
_____  1  TN      WS%%%%   ___                SYSSTC      RWSCTLR
 ____  1  TN      WS%%%%S  ___                OPS_HIGH    RWSSRVR
```

Creating clusters

A cluster is a set of application servers that you manage together as a way to balance workload.

Before you create a cluster determine:

- Whether you want enterprise bean requests routed to the node on which the client resides.
- If you want to use HTTP memory-to-memory replication.
- The configuration settings you want to specify for the first cluster member. A copy of the first cluster member that you create is stored as part of the cluster data and becomes the template for all additional cluster members that you create.
- The node on which the first cluster member resides.

Also review the content of “Clusters and workload management” on page 371, especially the information about setting cluster weights.

You might want to create a cluster if you need to:

- Balance your client requests across multiple application servers.
- Provide a highly available environment for your applications.

A cluster enables you to manage a group of application servers as a single unit, and distribute client requests among the application servers that are members of the cluster.

On the z/OS platform, if you plan to create a cluster of servers that spans multiple systems in a sysplex and has stateful session beans with an activation policy of Transaction deployed in them, the passivation directory should reside on an HFS (hierarchical file system) that is shared across the multiple systems in the sysplex on which the clustered servers are running.

For more information, see “Considerations for clustered servers and stateful session beans” on page 408.

To create a cluster:

1. In the administrative console, click **Servers** > **Clusters** > **New**. The Create a new cluster wizard starts.
2. Specify a name for the cluster.
3. **Optional:** Specify a short name for the cluster. This field only appears if you are running on a z/OS system.

   For clustered servers, the WLM application environment is the default value for the cluster short name. If you specify a short name for a cluster, the name:

   - Must be one to eight characters in length
   - Must contain only alpha-numeric or national language characters.
   - Cannot start with a number.
   - Must be unique in the cell
   - Cannot be the same as the value specified on the `ClusterTransitionName` custom property of any non-clustered server. Do not specify a cluster transition name for a server that is part of a cluster.
4. Select **Prefer local** if you want to enable node-scoped routing optimization. This option is enabled by default. When this option is enabled, if possible, EJB requests are routed to the client node. This option improves performance because client requests are sent to local enterprise beans.
5. Select **Configure HTTP session memory-to-memory replication** if you want a memory-to-memory replication domain created for this cluster. The replication domain is given the same name as the cluster and is configured with the default settings for a replication domain. When the default settings are in effect, a single replica is created for each piece of data and encryption is disabled. Also, the Web container for each cluster member is configured for memory-to-memory replication.

   To change these settings for the replication domain, click **Environment** > **Replication domains** > *replication_domain_name*. To modify the Web container settings, click **Servers** > **Clusters** > *cluster_name* > **Cluster members** > *cluster_member_name* > **Web container settings** > **Session management** > **Distributed environment settings** in the administrative console. If you change these settings for one cluster member, you might also need to change them for the other members of this cluster.
6. Click **Next**.
7. Choose whether to create an empty cluster or to create the first member of the cluster.

   If you decide to create an empty cluster, when you are ready to add members to this cluster, in the administrative console, click **Servers** > **Clusters** > *cluster_name* > **Cluster members** > **New**.

   To create an empty cluster:

   a. Select **None. Create an empty cluster.**

b. Click **Next** to display a summary of the defined cluster.

c. Click **Finish** to create the cluster, or click **Cancel** if you decide not to create this cluster.

When you create the first cluster member, remember that a copy of the first cluster member that you create is stored as part of the cluster data and becomes the template for all additional cluster members that you create.

a. Specify the name of the first cluster member.

b. Select the node on which you want this cluster member to reside.

c. Specify the weight value for the cluster member. The weight value controls the amount of work that is directed to the application server. If the weight value for this server is greater than the weight values that are assigned to other servers in the cluster, then this server receives a larger share of the workload. The weight value represents a relative proportion of the workload that is assigned to a particular application server. The value can range from `0` to `20`. See "Clusters and workload management" on page 371 for more information.

   On the z/OS platform, weight is used to balance some of the workload types, but others are balanced by the z/OS system.

   - For HTTP requests, weights are used to distribute HTTP traffic between the Web server plug-in and the controller handling the clustered application server. Assign a higher weight value to the application server that should receive the HTTP traffic.
   - For Web services calls, information is transferred from a servant in one application server to a controller in another application server. The application server that receives the call has the highest weight value.
   - Weight has no affect on Internet Inter-ORB Protocol (IIOP) requests. IIOP requests are distributed to the correct application server using the sysplex distributor.

d. Select **Generate unique HTTP ports** if you want to generate unique port numbers for every HTTP transport that is defined in the source server. When this option is selected, which is the default setting, this cluster member does not have HTTP transports or HTTP transport channels that conflict with any of the other servers that are defined on the same node. If you unselect this option, all of the cluster members will share the same HTTP ports.

e. Select the core group to which you want this cluster member to belong. You are prompted for the core group only if you have more than one core group defined for this cluster.

f. Select one of the following options as the basis for the first cluster member.

   - Create the member using an application server template.
   - Create the member using an existing application server as a template.
   - Create the member by converting an existing application server.

   Important: You can only add an existing application server to the cluster if you select that server as the first cluster member. You cannot add other existing application servers to that cluster after you create the first cluster member. If you add an existing server to a cluster, the only way to remove that server from the cluster is to delete the server. Therefore, you might want to use the existing server as a template for the first cluster member instead of as the cluster member. If you keep the original application server out of the cluster, you can reuse that server as the template if you need to rebuild the configuration.

8. Click **Next**.

9. Create additional cluster members. Before you create additional cluster members, check the configuration settings of the first cluster member. These settings are displayed at the bottom of the Create additional cluster members panel of the Create a new cluster wizard. For each additional member that you want to create:

   a. Specify a unique name for the member. The name must be unique within the node.

   b. Select the node to which you want to assign the cluster member.

c. Specify the weight you want given to this member. The weight value controls the amount of work that is directed to the application server. If the weight value for the server is greater than the weight values that are assigned to other servers in the cluster, then the server receives a larger share of the workload. The value can range from `0` to `20`.

d. Specifies a short name for this cluster member. The short name is the default z/OS job name and identifies the cluster member to the native facilities of the operating system, such as Workload Manager (WLM), Automatic Restart Manager, SAF (for example, RACF), started task control, and others.

e. Select **Generate unique HTTP ports** if you want to generate unique port numbers for every HTTP transport that is defined in the source server.

f. Click **Apply**. You can edit the configuration settings of any of the newly created cluster members other than the first cluster member, or you can create additional cluster members. Click **Previous** to edit the properties of the first cluster member.

10. When you finish creating cluster members, click **Next**.
11. View the summary of the cluster and then click **Finish** to create the cluster, click **Previous** to return to the previous wizard panel and change the cluster, or click **Cancel** to exit the wizard without creating the cluster.
12. To further configure a cluster, click **Servers > Clusters**, and then click the name of the cluster. Only the **Configuration** and **Local Topology** tabs appear until you save your changes.
13. Click **Review** to review your cluster configuration settings. Repeat the previous step if you need to make additional configuration changes.
14. If you do not want to make any additional configuration changes, select Synchronize changes with Nodes and then click **Save**. Your changes are saved and synchronized across all of your nodes.

    Important: If you click **Save**, but do not select Synchronize changes with Nodes, when you restart the cluster, WebSphere Application Server does not start the cluster servers because it cannot find them on the node. If you want to always synchronize your configuration changes across your nodes, you can select Synchronize changes with Nodes as one of your console preferences.
15. Restart the cluster.

You have a configured cluster to which you can assign work requests. The **Runtime** and **Local Topology** tabs appear the next time you access this page.

You can:

- Click **Servers > Clusters >** *cluster_name* to view or to change the configuration settings for a cluster. For example, if you are running in a high availability environment, you might want to select the **Enable failover of transaction log recovery** option for this cluster. This option allows the recovery of transactions to failover from one cluster member to another. See Chapter 8, “Balancing workloads with clusters,” on page 371 for more information about cluster configuration options.
- Create additional cluster members.
- Start the cluster.
- Use scripting to automate the task of creating clusters.

  See the *Administering applications and their environment* PDF for more information.
- Create a static routing table to temporarily handle IIOP routing for the cluster if your high availability infrastructure is disabled.

Creating a cluster: Basic cluster settings

Use this page to enter the basic settings for a cluster.

To view this administrative console page, click **Servers > Clusters > New**.

Cluster name

Specifies the name of the cluster. The cluster name must be unique within the cell.

Short name

Specifies the short name for this cluster. This field only appears if you are running on a z/OS platform.

The short name is the default z/OS job name and identifies the cluster to the native facilities of the operating system, such as Workload Manager (WLM), Automatic Restart Manager, SAF (for example, RACF), started task control, and others.

If you specify a short name for a cluster, the name:

- Must be one to eight characters in length. By default, WebSphere Application Server for z/OS assumes you are using a 7-character server short name (JOBNAME). If your naming standards require 8 characters, you can lengthen the 7-character server short name to 8 characters.
- Must contain only alpha-numeric or national language characters.
- Cannot start with a number.
- Must be unique in the cell

If you do not specify a short name, the system assigns a default short name that is automatically unique within the cell. You can change the generated short name to conform with your naming conventions.

Prefer local

Specifies that the node scoped routing optimization is enabled or disabled. The default is enabled, which means that Enterprise JavaBeans (EJB) requests are routed to the client node when possible. Enabling this setting improves performance because client requests are sent to local enterprise beans.

Configure HTTP session memory-to-memory replication

Specifies that when the cluster is created, a memory-to-memory replication domain is created for each of the members of this cluster.

If a replication domain is created, it is given the same name as the cluster and is configured with the default settings for a replication domain. When the default settings are in effect, a single replica is created for each piece of data and encryption is disabled. Also, the SIP container and Web container for each cluster member is configured for memory-to-memory replication.

To modify the replication domain settings, in the administrative console, click **Environment > Replication domains >** *replication_domain_name*.

The default mode setting for the replication domain is `Both client and server`. In this mode, all data sent to either the client or the server is replicated. This setting is good for an environment that has a middle to low traffic load. However, if your environment has a high traffic load, you should change the replication domain mode setting to either `Client only` , or `Server only`, because these settings provide better scaling. In `Client only` mode, only data sent to the client is replicated. In `Server only` mode, only data sent to the server is replicated.

To modify the mode setting for a replication domain, in the administrative console, click **Servers > Clusters >***cluster_name* **> Cluster members >** *cluster_member_name*, and then click either **SIP Container Settings** or **Web Container Settings**. Next click **Session management > Distributed environment settings**, and select a different mode. It does not matter whether you change the mode under the SIP container or the Web container because the same replication domain settings apply to both containers.

Important: If you change any of the replication domain settings for one cluster member, including the mode setting, you should change them for all of the other members of the cluster.

Creating a cluster: Create first cluster member

Use this page to specify settings for the first cluster member.

There are two ways to create the first member of a cluster:

- You can create the first member when you create a new cluster.

  To create a new cluster, in the administrative console, click **Servers** > **Clusters** > **New**.
- You can create an empty cluster and then add a first member after you finish creating the cluster.

  To create a cluster member for an existing cluster, in the administrative console, click **Servers > Clusters** > *cluster_name* > **Cluster members** > **New**.

When you create the first cluster member, a copy of that member is stored as part of the cluster data and becomes the template for all additional cluster members that you create.

When adding servers to a cluster, remember that the only way to remove an application server from a cluster is to delete the application server from the list of cluster members.

Member name

Specifies the name of the application server that is created for the cluster.

The member name must be unique on the selected node.

Select node

Specifies the node on which the application server resides.

Short name

Specifies the short name for this cluster member. This field only applies to the z/OS platform.

The short name is the default z/OS job name and identifies the cluster member to the native facilities of the operating system, such as Workload Manager (WLM), Automatic Restart Manager, SAF (for example, RACF), started task control, and others.

If you specify a short name for a cluster member, the name:

- Must be one to eight characters in length. By default, WebSphere Application Server for z/OS assumes you are using a 7-character server short name (JOBNAME). If your naming standards require 8 characters, you can lengthen the 7-character server short name to 8 characters.
- Must contain only alpha-numeric or national language characters.
- Cannot start with a number.
- Must be unique in the cell
- Cannot be the same as the value specified on the `ClusterTransitionName` custom property of any non-clustered server. Do not specify a cluster transition name for a server that is part of a cluster.

If you do not specify a short name, the system assigns a default short name that is automatically unique within the cell. You can change the generated short name to conform with your naming conventions.

Weight

Specifies the amount of work that is directed to the application server.

If the weight value for the server is greater than the weight values that are assigned to other servers in the cluster, the server receives a larger share of the cluster workload. The value can range from 0 to 20. Enter zero to indicate that you do not want requests to route to this application server unless this server is the only server that is available to receive requests.

On the z/OS platform, weight is used to balance some of the workload types, but others are balanced by the z/OS system.

- For HTTP requests, weights are used to distribute HTTP traffic between the Web server plug-in and the controller handling the clustered application server. Assign a higher weight value to the application server that should receive the HTTP traffic.
- For Web services calls, information is transferred from a servant in one application server to a controller in another application server. The application server that receives the call has the highest weight value.
- Weight has no affect on Internet Inter-ORB Protocol (IIOP) requests. IIOP requests are distributed to the correct application server using the sysplex distributor.

Core group

Specifies the core group in which the application server resides. This field displays only if you have multiple core groups configured. You can change this value only for the first cluster member.

Generate unique HTTP ports

Specifies that a unique HTTP port is generated for the application server. By generating unique HTTP ports for the application server, you avoid potential port collisions and configurations that are not valid.

Select basis for first cluster member:

Specifies the basis you want to use for the first cluster member.

If you select **Create the member using an application server template**, the settings for the new application server are identical to the settings of the application server template you select from the list of available templates.

If you select **Create the member using an existing application server as a template**, the settings for the new application server are identical to the settings of the application server you select from the list of existing application servers.

If you select **Create the member by converting an existing application server**, the application server you select from the list of available application servers becomes a member of this cluster.

If you select **None. Create an empty cluster** , a new cluster is created but it does not contain any cluster members.

Important: The basis options are available only for the first cluster member. All other members of a cluster are based on the cluster member template which is created from the first cluster member.

Creating a cluster: Summary settings

Use this administrative console page to view and save settings when you create a cluster or cluster member.

You can view this administrative console page whenever you create a new cluster or a new cluster member. This summary page displays your configuration changes before you commit the changes and the new cluster or cluster member is created.

To create a cluster, in the administrative console, click **Servers > Clusters > New**.

To create a cluster member for an existing cluster, in the administrative console, click **Servers > Clusters >** *cluster_name* **> Cluster members > New**

The bounding node group of the cluster is based on the first application server that is added as a member of the cluster. To select a different bounding node group, click **Servers > Clusters >** *cluster_name* in the administrative console and edit the settings for that cluster.

Review the changes to your configuration, and then click **Finish** to complete and save your work.

Creating a cluster: Create additional cluster members

Use this page to create additional members for a cluster. You can add a member to a cluster when you create the cluster or after you create the cluster. A copy of the first cluster member that you create is stored as part of the cluster data and becomes the template for all additional cluster members that you create.

To add members to a cluster, in the administrative console, click **Servers > Clusters >** *cluster_name* **> Cluster members > New**. After you enter the required information about the new cluster member, click **Add Member** to add this member to the cluster member list.

After adding a cluster member, you might need to change one or more of the property settings for this cluster member, or a another cluster member that you just added. To change one or more property settings for any cluster member that you just added, other than the first cluster member, select that 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 cluster member, select the member and then click **Delete**.

You cannot edit or delete the first cluster member or an already existing cluster member.

If you create additional cluster members immediately after you create the first cluster member, the list of cluster members includes a checklist in front of the names of these additional cluster members. However, a check box does not appear in front of the name of the first cluster member because you cannot delete this member or edit its settings. To modify the first cluster member, click **Previous**.

Similarly, if you are adding cluster members to a cluster that already has existing members, the existing members appear in the list of cluster members but a check box does not appear in front of the names of these cluster members. To delete one of these existing members or to change the settings of one of these cluster members, in the administrative console click **Servers > Clusters >** *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 application server that is created for the cluster.

The member name must be unique on the selected node.

Select node

Specifies the node on which the application server resides.

Short name

Specifies the short name for this cluster member. This field only applies to the z/OS platform.

The short name is the default z/OS job name and identifies the cluster member to the native facilities of the operating system, such as Workload Manager (WLM), Automatic Restart Manager, SAF (for example, RACF), started task control, and others.

If you specify a short name for a cluster member, the name:

- Must be one to eight characters in length. By default, WebSphere Application Server for z/OS assumes you are using a 7-character server short name (JOBNAME). If your naming standards require 8 characters, you can lengthen the 7-character server short name to 8 characters.
- Must contain only alpha-numeric or national language characters.
- Cannot start with a number.
- Must be unique in the cell

- Cannot be the same as the value specified on the `ClusterTransitionName` custom property of any non-clustered server. Do not specify a cluster transition name for a server that is part of a cluster.

If you do not specify a short name, the system assigns a default short name that is automatically unique within the cell. You can change the generated short name to conform with your naming conventions.

Weight

Specifies the amount of work that is directed to the application server.

If the weight value for the server is greater than the weight values that are assigned to other servers in the cluster, the server receives a larger share of the cluster workload. The value can range from 0 to 20. Enter zero to indicate that you do not want requests to route to this application server unless this server is the only server that is available to receive requests.

On the z/OS platform, weight is used to balance some of the workload types, but others are balanced by the z/OS system.

- For HTTP requests, weights are used to distribute HTTP traffic between the Web server plug-in and the controller handling the clustered application server. Assign a higher weight value to the application server that should receive the HTTP traffic.
- For Web services calls, information is transferred from a servant in one application server to a controller in another application server. The application server that receives the call has the highest weight value.
- Weight has no affect on Internet Inter-ORB Protocol (IIOP) requests. IIOP requests are distributed to the correct application server using the sysplex distributor.

Generate unique HTTP ports

Specifies that a unique HTTP port is generated for the application server. By generating unique HTTP ports for the application server, you avoid potential port collisions and configurations that are not valid.

Server cluster collection

Use this page to view information about and change configuration settings for a cluster. A cluster consists of a group of application servers. If one of the application servers fails, requests are routed to other members of the cluster.

To view this administrative console page, click **Servers > Clusters**.

To define a new cluster, click **New** to start the Create a new cluster wizard.

Name

Specifies a logical name for the cluster. The name must be unique among clusters within the containing cell.

Status

Specifies whether a cluster is stopped, partially started, started, or partially stopped.

If all cluster members are stopped, `stopped` displays for the status and state of the cluster. After you click **Start** or **Ripplestart** to start a cluster, the cluster state briefly displays as `starting`, and each server that is a member of that cluster launches if it is not already running. When the first member launches, the state changes to `partially started`. The state remains `partially started` until all cluster members are running. When all cluster members are running, the state changes to `running` and the status is `started`. Similarly, when you click **Stop** or **ImmediateStop** to stop a cluster, the state changes to `partially stopped` when the first member stops and then changes to `stopped` when all cluster members are not running.

Server cluster settings

Use this page to view or change the configuration of a server cluster instance, and to view the local topology of a server cluster instance.

To change the configuration and local topology of a server cluster, in the administrative console click **Servers > Clusters >** *cluster_name*.

To view runtime information, such as the state of the server cluster, click **Servers > Clusters >** *cluster_name*, and then click the **Runtime** tab.

To display the topology of a specific cluster, click **Servers > Clusters >** *cluster_name*, and then click the **Local Topology** tab.

If the high availability infrastructure is disabled and you require IIOP routing capabilities, click **Servers > Clusters >** *cluster_name*, then click on the **Runtime** tab, and then click **Export route table** to take a snapshot of the run-time cluster routing information as viewed by the Deployment Manager, and serialize it to the file system under the cluster's configuration directory. A static route table for IIOP cluster traffic is created. You should then force a node synchronize to ensure that the file is distributed to all of the nodes in the cell.

Because the information contained in the static route table does not account for server run-time state, you should only use this option if the high availability infrastructure is disabled.

Use of a static route table preempts the use of the dynamic routing table that is contained in cluster members. After the static file is transferred to a node, whenever a cluster member residing in that node starts, it uses the static table instead of the dynamic table to handle IIOP routing. If a cluster member is running when you create the static route table, you must restart that cluster member to give it access to the static route table information because the table is loaded at runtime.

After the table is created, an informational message, similar to the following message, is issued that indicates the name of the file that contains the table and where that file is located:

```
The route table for cluster MyCluster was exported to file
/home/myInstall/was/server/profiles/dmgrProfile/config/cells/
   MyCell/clusters/Myfile.wsrttble.
```

As this message illustrates, the file containing the table is placed in the `config` directory of the deployment manager for this cluster. You should keep a record of this location so that you can delete this file when you are ready to start using dynamic routing again.

Important: If you use this option, you must statically set the ORB_LISTENER_ADDRESS port on each of the cluster members because the route table is static and the cluster members do not communicate during state changes. If this port is not assigned, the cluster members restart on different ports and the static routing information is not able to route requests to the cluster members.

Cluster name:

Specifies a logical name for the cluster. The name must be unique among clusters within the containing cell.

Data type	String

Short name:

Specifies the short name for this cluster. This field only applies to the z/OS platform.

The short name is the default z/OS job name and identifies the cluster to the native facilities of the operating system, such as Workload Manager (WLM), Automatic Restart Manager, SAF (for example, RACF), started task control, and others.

If you specify a short name for a cluster, the name:

- Must be one to eight characters in length. By default, WebSphere Application Server for z/OS assumes you are using a 7-character server short name (JOBNAME). If your naming standards require 8 characters, you can lengthen the 7-character server short name to 8 characters.
- Must contain only alpha-numeric or national language characters.
- Cannot start with a number.
- Must be unique in the cell

If you do not specify a short name, the system assigns a default short name that is automatically unique within the cell. You can change the generated short name to conform with your naming conventions.

Data type	String

Unique Id:

Specifies the unique ID of this cluster.

The unique ID property is read-only. The system automatically generates the value.

Bounding node group name:

Specifies the node group that bounds this cluster. All application servers that are members of a cluster must be on nodes that are members of the same node group.

A node group is a collection of WebSphere Application Server nodes. A node is a logical grouping of managed servers, usually on a computer system that has a distinct IP host address. All application servers that are members of a 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 clusters formed across the nodes in the node group can host the same application in each 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.

Enable failover of transaction log recovery:

Specifies that for the transaction service component, failover of the transaction log for recovery purposes is enabled or disabled. The default is disabled.

When this setting is enabled, and the transaction service properties required for peer recovery of failed application servers in a cluster are properly configured, failover recovery of the transaction log occurs if the server processing the transaction log fails. If the transaction services properties required for peer recovery of failed application servers in a cluster are not properly configured, this setting is ignored.

State:

Specifies whether the cluster is stopped, starting, or running.

If all cluster members are stopped, the cluster state is *stopped*. After you request to start a cluster, the cluster state briefly changes to *starting* and each server that is a member of that cluster launches, if it is not already running. When the first member launches, the state changes to *websphere.cluster.partial.start*. The state remains *partially started* until all cluster members are running, then the state changes to *running*.

Similarly, when stopping a cluster, the state changes to *partially stopped* as the first member stops and changes to *stopped* when all members are not running.

Data type	String
Range	Valid values are starting, partially started, running, partially stopped, or stopped.

Cluster topology

Use this page to display, in a tree format, a list of all of the application server clusters defined for your WebSphere Application Server environment. The list shows all of the nodes and cluster members that are included in each cluster contained in a cell.

To view this page, in the administrative console, click **Servers > Cluster topology**.

Considerations for clustered servers and stateful session beans

For a cluster of servers that span multiple systems in a sysplex and will have stateful session beans with an activation policy of *Transaction* deployed in them, the passivation directory should reside on an HFS (hierarchical file system) that is shared across the multiple systems in the sysplex on which the clustered servers will be running.

Before creating the cluster, it is important to consider the following:

- Does the cluster have cluster members that are on different systems in the sysplex?
- If so, do any of the applications that are to be deployed or are already deployed have stateful session beans?
- If so, do the Stateful Session beans have an activation policy of Transaction?

If you answered yes to all the questions above, then you should define a shared HFS (hierarchical file system) to be used as the passivation directory for the stateful session beans.

The name of the passivation directory should contain the install root and cluster name. In the following example, /WebSphere/V6R0M0/AppServer is also known as USER_INSTALL_ROOT, and the name of the cluster is **cluster1**.

```
/WebSphere/V6R0M0/AppServer/passivation/cluster1
```

For optimal performance, you should create a passivation directory for each cluster. Also, the default passivation directory should not be used for clusters; it should be used for non-clustered servers and servers that do not have stateful session beans with an activation policy of *Transaction*.

For information about defining a shared HFS, see *z/OS UNIX System Services Planning*.

For information on specifying the passivation directory with the WebSphere administrative console, see the *Administering applications and their environment* PDF.

Enabling static routing for a cluster

If your high availability infrastructure is disabled and you require IIOP routing capabilities, you can create a static routing table for the members of a cluster to use to handle enterprise bean requests. Because the information contained in this static routing table does not account for server runtime state, you should delete this table and return to using the dynamic routing table as soon as your high availability infrastructure is enabled.

Before you create a static route table, ensure that the ORB_LISTENER_ADDRESS port is set on each of the cluster members. Because the route table you create is static, and the cluster members do not communicate during state changes, if you do not set the ORB_LISTENER_ADDRESS port on each of the cluster members, the cluster members restart on different ports and the deployment manager is not able to route IIOP requests to the cluster members.

You should only create a static route table if your high availability infrastructure is disabled and you require IIOP routing capabilities. To create a static route table:

1. In the administrative console, click **Servers > Clusters**.
2. Select the fully-configured cluster for which static routing is required and click **Start** if it is not already running.
3. Ensure that the cluster is in a fully available state such that a client can route across the entire cluster.
4. Click the name of the cluster, and then click **Export route table** to create a static route table for that cluster. After the table is created, an informational message, similar to the following message, is issued that indicates the name of the file that contains the table and where that file is located:

   ```
   The route table for cluster MyCluster was exported to file
   /home/myInstall/was/server/profiles/dmgrProfile/config/cells/MyCell/
        clusters/Myfile.wsrttble.
   ```

   As this message illustrates, the file containing the table is placed in the config directory of the deployment manager for that cluster. You should keep a record of this location so that you can delete this file when you are ready to start using dynamic routing again.
5. Click **Save** , and then click **Synchronize changes with Nodes**. This step forces a node synchronization which ensures that the file is distributed to all of the nodes in the cell.
6. Select the cluster again and click **Stop** to stop the cluster.
7. Select the cluster again and click **Start**. The cluster members do not start to use the static route table until you stop and then restart the cluster.

The cluster members use the static route table to perform IIOP routes.

When your high availability infrastructure is enabled, disable static routing so that the cluster members resume using dynamic routing:

1. Set the ORB_LISTENER_ADDRESS property back to 0 (zero).
2. Delete the static route table file from the config directory of the deployment manager for this cluster.
3. Force a node synchronize to ensure the file is removed from the nodes.
4. Stop the cluster and then start the cluster again.

Adding members to a cluster

You create an application server as a member of a configured cluster.

Create a cluster if you do not already have a configured cluster.

See “Creating clusters” on page 397 for more information.

If you are migrating from a previous version of WebSphere Application Server, you can upgrade a portion of the nodes in a cell, while leaving others at the other release level. For a time, you might be managing servers that are at a previous release level and servers that are running at the newer release in the same cell. In this mixed environment, you can only add cluster members for the WebSphere Application Server versions that already exist in the cluster. For example, if a cluster contains a WebSphere Application Server v5.x member, you can create a new v5.x member in that cluster. However, if a cluster does not contain a WebSphere Application Server v5.x member, you cannot create a new v5.x member in that cluster.

To create a new cluster member, view information about existing cluster members, or manage existing cluster members:

1. In the administrative console, click **Servers > Clusters >** *cluster_name* > **Cluster members**. The Cluster members page lists members of a cluster, and for each member indicates:
   - The node on which the member resides.
   - The version of the application server. This information shows whether the cluster is a mixed cluster.
   - The configured weight for the member.
   - The runtime weight for the member. This weight indicates the proportionate workload that is currently directed to this cluster member.
   - Whether the member is started, stopped, or encountering problems.
2. Click **New** to create a new cluster member.

   Clicking **New** starts the Create a new cluster member wizard. This wizard allows you to add new members to an already configured cluster.

   A copy of the first cluster member that you create is stored as part of the cluster data and becomes the template for all additional cluster members that you create.

   a. Specify a name for the cluster member (application server). The name must be unique within the node.

   b. Select the node for the cluster member.

   c. Specify the server weight. The weight value controls the amount of work that is directed to the application server. If the weight value for the server is greater than the weight values that are assigned to other servers in the cluster, then the server receives a larger share of the workload. The value can range from `0` to `20`.

      On the z/OS platform, weight is used to balance some of the workload types, but others are balanced by the z/OS system.

      - For HTTP requests, weights are used to distribute HTTP traffic between the Web server plug-in and the controller handling the clustered application server. Assign a higher weight value to the application server that should receive the HTTP traffic.
      - For Web services calls, information is transferred from a servant in one application server to a controller in another application server. The application server that receives the call has the highest weight value.
      - Weight has no affect on Internet Inter-ORB Protocol (IIOP) requests. IIOP requests are distributed to the correct application server using the sysplex distributor.

   d. Specify whether to generate unique HTTP ports.

   e. Click **Add Member** to finish defining the cluster member. The first cluster member for this cluster is used as the template for this cluster member. You can repeat these steps to define other cluster members.

   f. When you finish defining additional cluster members, review the summary information for the new cluster members. If you have to change any of the property settings for any of the new members, select that cluster member, and then click **Edit**. When you finish changing the property settings, click **Update Member** to save your changes.

   g. When you finish defining new cluster members, click **Next** to view the summary page for the cluster, and then click **Finish** to create these new cluster members.
3. Click **Review**, select Synchronize changes with Nodes, and then click **Save** to save your changes.

You created application servers that are members of an existing server cluster.

If when you created the new members, you chose to generate unique ports, update the alias list for the virtual host that you plan to use with the new servers.

You can:

- Click on the name of the cluster member under **Member name** on the Cluster members page in the administrative console, and examine the settings for a specific cluster member.

- Click **Servers > Application Servers >** *cluster_member_name* or click **Servers > Clusters >** *cluster_name* **> Cluster members >** *cluster_member_name* to specify additional application server properties for a cluster member.
- Start the cluster. See "Starting clusters" on page 414 for more information.
- Use scripting to automate the task of adding cluster members.

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

Cluster member collection

Use this page to view and manage application servers that belong to a cluster.

You can also use this page to change the weight of any of the listed application servers.

A copy of the first cluster member that you create is stored as part of the cluster data and becomes the template for all additional cluster members that you create.

Any individual configuration change that you make to a cluster member does not affect the configuration settings of the cluster member template. You must use wsadmin commands to modify this template. Similarly, any changes that you make to the template does not affect existing cluster members.

See the *Using the administrative clients* PDF for more information on how to modify this template.

To view this administrative console page, click **Servers > Clusters >** *cluster_name* **> Cluster members**.

Member name

Specifies the name of the server in the cluster. On most platforms, the name of the server is the process name. The name must match the (object) name of the application server.

Node

Specifies the name of the node for the cluster member.

Version

Specifies the version of the WebSphere Application Server product on which the cluster member runs.

Configured weight

Specifies the weight that is currently configured for the cluster member. The weight determines the amount of work that is directed to the application server. If the weight value for the server is greater than the weight values assigned to other servers in the cluster, the server receives a larger share of the cluster workload.

To change the configured weight for a cluster member you can either specify a new weight in the Configured weight field and click the **Update** button for the Configured weight column, or click on the name of the cluster member. Clicking the name of the cluster member navigates you to the page where you can change any of the configuration settings for that cluster member.

Runtime weight

Specifies the proportionate workload that is currently directed to the cluster member in comparison to the configured weight for that cluster member. To change the proportion, specify a new weight in the Runtime weight field and click the **Update** button for the Runtime weight column.

Status

Specifies whether a cluster member is running, stopped, or unavailable.

If a cluster member is stopped, its status is `Stopped`. After you request to start a cluster member by clicking **Start**, the status becomes `Started`. After you click **Stop**, its status changes to `Stopped` when it stops running.

Note that 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 cluster member.

Cluster member settings

Use this page to manage the members of a cluster. A cluster of application servers are managed together and participate in workload management.

A copy of the first cluster member that you create is stored as part of the cluster data and becomes the template for all additional cluster members that you create.

Any individual configuration change that you make to a cluster member does not affect the configuration settings of the cluster member template. You must use wsadmin commands to modify this template. Similarly, any changes that you make to the template does not affect existing cluster members. See the *Using the administrative clients* PDF for more information on how to modify this template.

To view this administrative console page, click **Servers > Clusters >** *cluster_name* **> Cluster members >** *cluster_member_name*.

Member name:

Specifies the name of the server in the cluster. On most platforms, the name of the server is the process name. The name must match the (object) name of the application server.

Data type	String

Weight:

Specifies the amount of work that is directed to the application server. If the weight value for the server is greater than the weight values assigned to other servers in the cluster, then the server receives a larger share of the server workload.

On the z/OS platform, weight is used to balance some of the workload types, but others are balanced by the z/OS system.

- For HTTP requests, weights are used to distribute HTTP traffic between the Web server plug-in and the controller handling the clustered application server. Assign a higher weight value to the application server that should receive the HTTP traffic.
- For Web services calls, information is transferred from a servant in one application server to a controller in another application server. The application server that receives the call has the highest weight value.
- Weight has no affect on Internet Inter-ORB Protocol (IIOP) requests. IIOP requests are distributed to the correct application server using the sysplex distributor.

Data type	Integer
Range	0 to 20

Unique ID:

Specifies a numerical identifier for the application server that is unique within the cluster. The ID is used for affinity.

Data type	Hexadecimal

Short name:

Specifies the short name for this cluster member. This field only applies to the z/OS platform.

The short name is the default z/OS job name and identifies the cluster member to the native facilities of the operating system, such as Workload Manager (WLM), Automatic Restart Manager, SAF (for example, RACF), started task control, and others.

If you specify a short name for a cluster member, the name:

- Must be one to eight characters in length. By default, WebSphere Application Server for z/OS assumes you are using a 7-character server short name (JOBNAME). If your naming standards require 8 characters, you can lengthen the 7-character server short name to 8 characters.
- Must contain only alpha-numeric or national language characters.
- Cannot start with a number.
- Must be unique in the cell
- Cannot be the same as the value specified on the `ClusterTransitionName` custom property of any non-clustered server. Do not specify a cluster transition name for a server that is part of a cluster.

If you do not specify a short name, the system assigns a default short name that is automatically unique within the cell. You can change the generated short name to conform with your naming conventions.

Data type	String

Run in development mode:

Enabling this option may reduce the startup time of an application server. This may include JVM settings such as disabling bytecode verification and reducing JIT compilation costs. Do not enable this setting on production servers. This setting is only available on application servers running WebSphere Application Server Version 6.0 and later.

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 will not be started in development mode. Setting this option to `true` specifies that the server will be started in development mode (with settings that will speed server startup time).

Data type	Boolean
Default	false

Parallel start:

Specifies whether to start the server on multiple threads. Starting the server on multiple threads might shorten the startup time.

Specifies that you want the server components, services, and applications to start in parallel rather than sequentially.

The default setting for this option is `true`, which indicates that the server be started using multiple threads. Setting this option to `false` specifies that the server will not be started in using multiple threads (which may lengthen startup time).

The order in which applications start depends on the weight you assign to each application. The application with the lowest starting weight starts first. Applications with the same starting weight start in parallel. You use the *Starting weight* field on the **Applications > Enterprise Applications >** *application_name* page of the administrative console to set the starting weight for an application.

Data type	Boolean
Default	true

Class loader policy:

Specifies whether there is a single class loader to load all applications or a different class loader for each application.

Class loading mode:

Specifies whether the class loader should search in the parent class loader or in the application class loader first to load a class. The standard for Developer Kit class loaders and WebSphere Application Server class loaders is `Parent first`.

This field only applies if you set the Class loader policy field to `single`.

If you select `Parent last`, your application can override classes contained in the parent class loader, but this action can potentially result in ClassCastException or linkage errors if you have mixed use of overridden classes and non-overridden classes.

Process Id:

Specifies the native operating system process ID for this server.

The process ID property is read only. The system automatically generates the value.

Cell name:

Specifies the name of the cell in which this server is running.

The Cell name property is read only.

Node name:

Specifies the name of the node in which this server is running.

The Node name property is read only.

State:

Specifies the run-time execution state for this server.

The State property is read only.

Starting clusters

You can start all members of a cluster at the same time by requesting that the state of a cluster change to *running*. That is, you can start all application servers in a server cluster at the same time.

Make sure that the members of your cluster have the debug port properly set. If multiple servers on the same node have the same debug port set, the cluster could fail to start. See “Java virtual machine settings” on page 351 for more information on how to change the debug port.

When you request that all members of a cluster start, the cluster state changes to *partially started* and each server that is a member of that cluster launches, if it is not already running. After all members of the cluster are running, the cluster state becomes *running*.

Important: From the z/OS MVS console, you must individually start each server that you want to run. With the administrative console, you can start each server individually or start a cluster which automatically starts all servers defined as part of that cluster.

1. In the administrative console, click **Servers > Clusters**.
2. Select the clusters whose members you want started.
3. Click **Start** or **RippleStart**.
   - **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.

   Important: If recently added cluster members do not start, you might not have selected Synchronize changes with Nodes when you added the members to the cluster. To determine if this is the problem:

   a. In the administrative console click **Servers > Clusters**, select the cluster whose members did not start, and click **Stop**.
   b. Click the name of the cluster and then click **Review**
   c. Select Synchronize changes with Nodes, and then click **Save**.
   d. Start the cluster and verify that all of the cluster members now start.

When you start the members of a cluster, you automatically enable workload management.

See Chapter 8, “Balancing workloads with clusters,” on page 371 for more information about the tasks that you can complete with clusters.

Stopping clusters

Use this task to stop a cluster and any application servers that are members of that cluster.

You can stop all application servers that are members of the same cluster at the same time by stopping the cluster.

1. Click **Servers > Clusters** in the console navigation tree to access the Server Cluster page.
2. Select those clusters whose members you want stopped.
3. Click **Stop** or **Immediate Stop**.
   - **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.
   - **Immediate Stop** brings down 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.

All application servers in the sysplex associated with this cluster are issued a request to stop. In addition, a **stop** can be issued against each individual server from the MVS console. To shut down the WebSphere Application Server for z/OS environment on a system, stop that system daemon. Stopping the system daemon brings down all other server instances on the system. To bring WebSphere for z/OS down on all systems, stop the daemons on all systems. When you stop the location service daemon on one system, it does not bring down the servers on the other systems.

See Chapter 8, “Balancing workloads with clusters,” on page 371 for more information about the tasks you can complete with clustering.

Replicating data across application servers in a cluster

Use this task to configure a data replication domain to transfer data, objects, or events for session manager, dynamic cache, or stateful session beans. Data replication domains use the data replication service (DRS), which is an internal WebSphere Application Server component that performs replication services, including replicating data, objects, and events among application servers.

If you configured a data replication domain with a previous version of WebSphere Application Server, you might be using a multi-broker replication domain. Any replication domains that you create with the current version of WebSphere Application Server are data replication domains. Migrate any multi-broker replication domains to data replication domains. To learn the differences between the two types of replication domains, see “Comparison of multi-broker versus data replication domains” on page 421 and “Migrating servers from multi-broker replication domains to data replication domains” on page 419.

Use this task to configure *replication*, a service that transfers data, objects, or events among the application servers in a cluster. Use replication to prevent loss of session data with session manager, to further improve the performance of the dynamic cache service, and to provide failover in stateful session beans. For more information about replication, see “Replication” on page 417.

Important: If you select the **Configure HTTP memory-to-memory replication** option when you create a cluster, the replication domain is automatically created for you.

1. Create a replication domain. Use one of the following methods to create a replication domain:
   - **Create a replication domain manually.**

     To create a replication domain manually without creating a new cluster, click **Environment > Replication domains > New** in the administrative console.

     On this page you can specify the properties for the replication domain, including timeout, encryption, and number of replicas. See “Data replication domain settings” on page 418 for more information about the properties that you can configure for your replication domain.
   - **Create a replication domain when you create a cluster.**

     To create a replication domain when you create a cluster, click **Servers > Clusters > New** in the administrative console. Then click **Configure HTTP memory-to-memory replication**. The replication domain that is created has the same name as the cluster and has the default settings for a replication domain. The default settings for a replication domain are to create a single replica of each piece of data and to have encryption disabled. To modify the replication domain properties, click **Environment > Replication domains >** *replication_domain_name* in the administrative console. See “Creating clusters” on page 397 for more information about creating a cluster.

   For more information about the replication domain settings that you can configure in the administrative console, see “Data replication domain settings” on page 418
2. Configure the consumers, or the components that use the replication domains. Dynamic cache, session manager, and stateful session beans are the three types of replication domain consumers. Each type of consumer must be configured with a different replication domain. For example, session manager uses one domain and dynamic cache uses a different replication domain. However, use one replication domain if you are configuring HTTP session memory-to-memory replication and stateful session bean replication. Using one replication domain in this case ensures that the backup state information of HTTP sessions and stateful session beans are on the same application servers.

- To configure dynamic cache replication, see the *Administering applications and their environment* PDF.
- To configure memory-to-memory replication for session manager, see the *Administering applications and their environment* PDF.
- To configure replication of stateful session beans, see the *Administering applications and their environment* PDF.

Data is replicating among the application servers in a configured replication domain.

If you select DES or 3DES as the encryption type for a replication domain, an encryption key is used for the encryption of messages. At regular intervals, for example once a month, you should go to the **Environment > Replication domains** page in the administrative console, and click **Regenerate encryption key** to regenerate the key. After the key is regenerated, you must restart all of the application servers that are configured as part of the replication domain. Periodically regenerating the key improves data security.

Replication

Replication is a service that transfers data, objects, or events among application servers. Data replication service (DRS) is the internal WebSphere Application Server component that replicates data.

Use data replication to make data for session manager, dynamic cache, and stateful session beans available across many application servers in a cluster. The benefits of using replication vary depending on the component that you configure to use replication.

- Session manager uses the data replication service when configured to do memory-to-memory replication. When memory-to-memory replication is configured, session manager maintains data about sessions across multiple application servers, preventing the loss of session data if a single application server fails. For more information about memory-to-memory replication, see the *Administering applications and their environment* PDF.
- Dynamic cache uses the data replication service to further improve performance by copying cache information across application servers in the cluster, preventing the need to repeatedly perform the same tasks and queries in different application servers. For more information about replication in the dynamic cache, see the *Administering applications and their environment* PDF.
- Stateful session beans use the replication service so that applications using stateful session beans are not limited by unexpected server failures. For more information about stateful session bean failover, see the *Developing and deploying applications* PDF.

You can define the number of *replicas* that DRS creates on remote application servers. A replica is a copy of the data that copies from one application server to another. The number of replicas that you configure affects the performance of your configuration. Smaller numbers of replicas result in better performance because the data does not have to copy many times. However, if you create more replicas, you have more redundancy in your system. By configuring more replicas, your system becomes more tolerant to possible failures of application servers in the system because the data is backed up in several locations.

By having a single replica configuration defined, you can avoid a single point of failure in the system. However, if your system must be tolerant to more failure, introduce extra redundancy in the system. Increase the number of replicas that you create for any HTTP session that is replicated with DRS. Any replication domain that is used by dynamic cache must use a full group replica.

Session manager, dynamic cache, and stateful session beans are the three *consumers* of replication. A consumer is a component that uses the replication service. When you configure replication, the same types of consumers belong to the same *replication domain*. For example, if you are configuring both session manager and dynamic cache to use DRS to replicate objects, create separate replication domains for each consumer. Create one replication domain for all the session managers on all the application servers and one replication domain for the dynamic cache on all the application servers. The only exception to this rule is to create one replication domain if you are configuring replication for HTTP sessions and stateful session beans. Configuring one replication domain in this case ensures that the backup state information is located on the same backup application servers.

See the *Administering applications and their environment* PDF for more information on how to configure replication.

Replication domain collection

Use this page to view the configured replication domains that are used for replication by the HTTP session manager, dynamic cache service, and stateful session bean failover components. All components that need to share information must be in the same replication domain. Data replication domains replace multi-broker replication domains that were available for replication in prior releases. Migrated application servers use multi-broker replication domains which are collections of replicators. You should migrate any multi-broker replication domains to be data replication domains.

To view this administrative console page, click **Environment > Replication domains**.

Name

Specifies a name for the replication domain. The name of the replication domain must be unique within the cell.

Domain type

Following are the two types of replication domains:

Multi-broker domain	Specifies a replication domain that was created with a previous version of WebSphere Application Server. This type of replication domain consists of replicator entries. Support of this type of domain remains for backward compatibility, but is deprecated. Multi-broker and data replication domains do not communicate with each other, so migrate any multi-broker replication domains to the new data replication domains. You cannot create a multi-broker domain or replicator entries in the administrative console after the deployment manager is upgraded to the current version of WebSphere Application Server.
Data replication domain	Specifies a replication domain created with the latest version of WebSphere Application Server. If the deployment manager has been upgraded to the latest version of WebSphere Application Server, you can create data replication domains only. With the data replication domain, you can specify a number of replicas instead of statically partitioning your replication settings. Specify a data replication domain for each consumer of the domain, for example, two separate domains for dynamic cache and session manager.

Data replication domain settings

Use this page to configure a data replication domain. Use data replication domains to transfer data, objects, or events for session manager, dynamic cache, or stateful session beans among the application servers in a cluster.

To view this administrative console page, click **Environment > Replication domains >** *replication_domain_name*.

Name:

Specifies a name for the replication domain. The name must be unique within the cell.

Request timeout:

Specifies how long a replication domain consumer waits when requesting information from another replication domain consumer before it gives up and assumes the information does not exist.

Units	seconds
Default	5 seconds

Encryption type:

Specifies the type of encryption to use when transferring replicated data to another area of the network. Select NONE if you don't want to use encryption, DES if you want to use data encryption standard, or 3DES if you want to use triple DES. The default is NONE. The DES and 3DES options encrypt data sent between application server processes (for example, session manager and dynamic caching). Encrypting data improves the security of the network that joins the processes.

If you select DES or 3DES, after you click **Apply** or **OK**, a key for global data replication is generated. At regular intervals, for example once each month, you should navigate to this page in the administrative console and click **Regenerate encryption key** to regenerate this key. Periodically regenerating the key enhances security.

Data type	String
Default	NONE

Number of replicas:

Specifies the number of replicas that are created for every entry or piece of data that is replicated in the replication domain.

Single replica	One replica is created. This is the default value.
Full group replica	Each object is replicated to every application server that is configured as a consumer of the replication domain.
Specific number of replicas	A custom number of replicas for any entry that is created in the replication domain.

Migrating servers from multi-broker replication domains to data replication domains

Use this task to migrate multi-broker replication domains to data replication domains. Any multi-broker domains that exist in your WebSphere Application Server environment were created with a previous version of WebSphere Application Server.

For HTTP session affinity to continue working correctly when migrating V5.x application servers to V6.1 application servers, you must upgrade all of the Web server plug-ins for WebSphere Application Server to the latest version before upgrading the application servers that perform replication.

After you upgrade your deployment manager to the latest version of WebSphere Application Server, you can create data replication domains only. Any multi-broker domains that you created with a previous release of WebSphere Application Server are still functional, however, you cannot create new multi-broker domains or replicators with the administrative console.

The different versions of application servers cannot communicate with each other. When migrating your servers to the current version of WebSphere Application Server, keep at least two application servers running on the previous version so that replication remains functional.

Perform this task on any multi-broker domains in your configuration after all of your servers that are using this multi-broker domain have been migrated to the current version of WebSphere Application Server. For more information about the differences between multi-broker domains and the data replication domains, see “Comparison of multi-broker versus data replication domains” on page 421.

The following examples illustrate the migration process for common configurations:

Migrating an application server configuration that uses an instance of data replication service in peer-to-peer mode

Use this migration path to migrate a replication domain that uses the default peer-to-peer configuration. Dynamic cache replication domains use the peer-to-peer topology.

Before you begin, migrate all the Web server plug-ins for your application server cluster to the current version.

1. Migrate one or more of your existing servers to the current version of WebSphere Application Server.
2. In the administrative console, create an empty data replication domain. Click **Environment > Replication domains > New** in the administrative console.
3. Add your migrated application servers to the new data replication domain. For example, if you are migrating 4 servers, migrate 2 servers first and add them to the new replication domain. Configure the servers to use the new domain by configuring the consumers of the replication domain.
4. When the new data replication domains are successfully sharing data, migrate the rest of the servers that are using the multi-broker replication domain to data replication domains.
5. Delete the empty multi-broker replication domain.

Migrating an application server configuration that uses an instance of the data replication service in client/server mode

Use this set of steps to migrate a replication domain that uses client/server mode.

Before you begin migrating a client/server mode replication domain, consider if migrating your replication domains might cause a single point of failure. Because you migrate the servers to the new type of replication domain one at a time, you risk a single point of failure if there are 3 or fewer application servers. Before migrating, configure at least 4 servers that use multi-broker replication domains. Perform the following steps to migrate the multi-broker domains to data replication domains:

1. Migrate one or more of your existing servers to the current version of WebSphere Application Server.
2. In the administrative console, click **Environment > Replication domains > New** to create an empty data replication domain.
3. Add your migrated servers to the new data replication domain. For example, if you are migrating 4 servers, migrate two of these servers and then add them to the new replication domain. Configure the servers to use the new domain by configuring the consumers of the replication domain.
4. Add a part of the clients to the new data replication domain.
5. When the new data replication domains are successfully sharing data, migrate the rest of the clients and servers that are using the multi-broker replication domain to data replication domains.
6. Delete the empty multi-broker replication domain.

Migrating a replication domain that uses HTTP session memory-to-memory replication that is overloaded at the application or web module level

1. Upgrade your deployment manager to the current version of WebSphere Application Server. All the application servers remain configured with the old multi-broker domains on the previous version of WebSphere Application Server.
2. In the administrative console, create an empty data replication domain. Click **Environment > Replication domains > New** in the administrative console.
3. Migrate each application server to the current version of WebSphere Application Server, one at a time. The remaining servers on the previous version of WebSphere Application Server can still communicate with each other, but not with the migrated servers. The migrated servers can also communicate with each other.
4. Continue migrating all of the servers to the current version of WebSphere Application Server. All of the application servers are still using multi-broker replication domains, so the features of data replication domains cannot be used.
5. Configure all of the application servers to use the new data replication domain, adding the application servers to the empty replication domain that you created.
6. Restart all of the application servers in the cluster.
7. Delete the empty multi-broker replication domain.

During this process, you might lose existing sessions. However, the application remains active through the entire process, so users do not experience down time during the migration. Create a new replication domain for each type of consumer. For example, create one replication domain for session manager and another replication domain for dynamic cache.

Comparison of multi-broker versus data replication domains

Data replication domains replace multi-broker domains for data replication between application servers in a cluster.

transition:
Any replication domains that are created with a previous version of WebSphere Application Server might be multi-broker domains. Migrate any multi-broker domains to the new data replication domains. Although you can configure existing multi-broker domains with the current version of WebSphere Application Server, after you upgrade your deployment manager, you can create only data replication domains in the administrative console.

Multi-broker and data replication domains both perform the same function, which is to replicate data across the consumers in a replication domain. Configure all the instances of replication that need to communicate in the same replication domain. You can also configure the session manager with both types of replication domains to use topologies such as peer-to-peer and client/server to isolate the function of creating and storing replicas on separate application servers. You can control the redundancy of replication for each type of replication domain. With a data replication domain, you can specify a specific number of replicas.

If you used multi-broker domains with earlier releases of WebSphere Application Server, use the following comparison chart to learn the differences between how V5.x and V6.x application servers use the two types of replication domains:

	V5.x application servers using replication domains	**V6.x application servers using replication domains**
Replication domain types	Uses only multi-broker replication domains for replication.	Servers that are using the current version of WebSphere Application Server can be configured to use both multi-broker replication domains and data replication domains for replication. The two types of domains provide backward compatibility with multi-broker domains that were created with a V5.x server. You should migrate any multi-broker domains to data replication domains.
Data transport method	Uses multi-broker domain objects that contain configuration information for the internal Java Message Service (JMS) provider, which uses JMS brokers as replicators.	Uses data replication domain objects that contain configuration information to configure the high availability framework on WebSphere Application Server. The transport is no longer based on the JMS API. Therefore, no replicators and no JMS brokers exist. You do not have to perform the complex task of configuring local, remote, and alternate replicators. The earlier version of WebSphere Application Server did not support data replication domains. The current version of WebSphere Application Server can be configured to perform replication using old multi-broker domains by ignoring any JMS-specific configuration and by using the other parameters to configure replication through the high availability framework.
Replication domain configuration	The earlier version of WebSphere Application Server encourages the sharing of replication domains between different consumers, such as session manager and dynamic cache.	The current version of WebSphere Application Server encourages creating a separate replication domain for each consumer. For example, create one replication domain for session manager and another replication domain for dynamic cache. The only situation where you should configure one replication domain is when configuring session manager replication and stateful session bean failover. Using one replication domain in this case ensures that the backup state information of HTTP sessions and stateful session beans are on the same application servers.

	V5.x application servers using replication domains	V6.x application servers using replication domains
Partial partitioning	You can configure partial partitioning. Partition the replication domain to filter the number of processes to send data.	Partial partitioning is deprecated. When using data replication domains, you can specify a specific number of replicas for each entry. However, if you specify a number of replicas larger than the number of backup application servers that are running, the number of replicas is the number of application servers that are running. After the number of application servers increases above your configured number of replicas, the number of replicas that are created is equal to the number that you specified.
Domain sharing	Multiple data replication service (DRS) instances share multi-broker domains. A limitation exists on the number of multi-broker domains that you can create because every multi-broker domain contains at least one replicator. A maximum of one replicator can be on each application server.	All DRS instances in a replication domain use the same mode. Each replication domain must contain either client only and server only instances, or client and server instances only. For example, if one instance is configured to client and server, all other instances must be client and server. If one instance in a replication domain is configured to be a client only, you can add client only and server only instances, but not a client and server instance.

To migrate multi-broker domains to data replication domains, see the *Migrating, coexisting, and interoperating* PDF.

Replicating data with a multi-broker replication domain

Use this task to mange replication domains that you migrated from a WebSphere Application Server v5 environment.

Multi-broker replication domains are not created in WebSphere Application Server v6.x environments. However they can be migrated from existing WebSphere Application Server v5.x environments. If you migrate v5.x multi-broker replication domains, you can use the Multi-broker domain panel in the v6.x administrative console to managed these domains.

Although you can manage migrated multi-broker domains with the current version of WebSphere Application Server, after you upgrade your deployment manager, you can create only data replication domains in the administrative console. Consider migrating any existing multi-broker domains to the new data replication domains. See “Migrating servers from multi-broker replication domains to data replication domains” on page 419 and “Multi-broker replication domains” on page 424 for more information about the benefits of migrating your replication domains.

If you do not have any existing replication domains, see “Replicating data across application servers in a cluster” on page 416 for information about creating new data replication domains.

If you are performing this task, it is assumed that you configured replication with a previous version of WebSphere Application Server and defined replication domains that list connected replicator entries (residing in managed servers in the cell) that can exchange data. You can manage these existing replication domains and replicator entries, but you cannot create new multi-broker replication domains or new replicator entries in the administrative console.

A replicator does not need to run in the same process as the application server that uses it. However, it might be easier to manage replicators and replication domains if a one-to-one relationship exists between replicators and application servers. During configuration, you can select the local replicator as the default replicator.

1. Manage multi-broker replication domain configuration settings. In the administrative console, click **Environment > Replication domains**.
2. Click on a **Multi-broker domain**, and update the values for a particular multi-broker replication domain. The default values are generally sufficient, especially for the pooling and timeout values.
   a. Name the replication domain.
   b. Specify the timeout interval.
   c. Specify the encryption type. The DES and TRIPLE_DES options encrypt data sent between WebSphere Application Server processes and better secure the network joining the processes.
   d. Partition the replication domain to filter the number of processes to which data is sent. Partitioning the replication domain is most often done if you are replicating data to support retrieval of an HTTP session if the process maintaining the HTTP session fails. Partitioning is not supported for sharing of cached data that is maintained by Web container dynamic caching.
   e. Specify whether you want a single replication of data to be made. Enable the option if you are replicating data to support retrieval of an HTTP session if the process maintaining the HTTP session fails.
   f. Specify whether processes should receive data in objects or bytes. Processes receiving data in objects receive the data and class definitions. Processes receiving data in bytes receive the data only.
   g. Configure a pool of replication resources. Pooling replication resources can enhance the performance of the replication service.
3. Maintain the replicators that you have already defined. You cannot create any new replicators. The default convention is to define a replicator in each application server that uses replication. However, you can define a pool of replicators, separate from the servers hosting applications.
   a. In the administrative console, click **Environment > Replication domains >** *replication_domain_name* **> Replicator entries >** *replicator_entry_name*.
   b. Specify a replicator name and select a server available within the cell to which you can assign a replicator. Also specify a host name and ports. Note that a replicator has two ports (replicator and client ports) that use the same host name but have different ports.
4. If you use the DES or TRIPLE_DES encryption type for a replicator, click **Regenerate encryption key** on the settings for a replication domain instance at regular intervals, such as monthly.

   Periodically changing the key enhances security.

Multi-broker replication domains

A multi-broker replication domain is a collection of replication entries, or *replicator* instances, used by clusters or individual servers within a cell. Multi-broker replication domains were created with a previous release of WebSphere Application Server.

Note: After you upgrade your deployment manager to the latest version of WebSphere Application Server, you can create data replication domains only. Any multi-broker domains that you created with a previous release of WebSphere Application Server are still functional, however, you cannot create new multi-broker domains or replicator instances with the administrative console. See “Comparison of multi-broker versus data replication domains” on page 421 for more information.

A replication entry, or replicator, is a run-time component that handles the transfer of internal WebSphere Application Server data. All replicators within a replication domain connect with each other, forming a network of replicators.

Components such as session manager and dynamic cache can connect to any replicator within a domain to receive data from their peer components on other application servers that are connected other replicators in the same domain. If the replicator that a component is connected to fails, the component automatically attempts to reconnect to another replicator in the domain and recover any data that was missed while the component was not connected to a replicator.

The default is to define a replication domain for a cluster when creating the cluster. However, replication domains can span across clusters.

Global default settings apply to a given replication domain across a cell. Most default settings tune and control the behavior of replicator entries that are in managed servers across the cell. Such default settings control the use of encryption or the serialization and transferring of objects. Some default settings tune and control how specific WebSphere Application Server functions (for example, session manager and dynamic caching) leverage replication, such as session use of partitions.

For situations that require settings values other than the default, change the values for a given replication domain. Settings include various resource allocation, replication strategies (such as grouping or partitioning) and methods, as well as some security related items.

If you are using replication for HTTP session failover, you might also need to filter where the session replicates. For example, only replicate to two places out of many. The global default settings define the partition size or number of groups and the session manager settings define the groups to which a particular instance belongs.

Filtering is less important if you are using replication to distribute information on data that is no longer valid and actual cached data maintained by dynamic caching. Replication does not occur for failover as much as for data synchronization across a cluster or cell when you likely want to avoid expensive costs for generating data potentially needed across those various servers.

Note that you can filter or segment by using multiple replication domains.

Multi-broker replication domain settings

Use this page to configure a multi-broker replication domain. This administrative console page applies only to replication domains that were created with a previous version of WebSphere Application Server. Replication domains use the data replication service (DRS).

To view this administrative console page, click **Environment > Replication domains >** *multibroker_replication_domain_name*.

An application server that is connected to a replicator within a domain can access the ame set of data sent out by any application server connected to any other replicator (including the same replicator). Data is not shared across replication domains.

Name:

Specifies a name for the replication domain. The name must be unique within the cell.

Request timeout:

Specifies the number of seconds that a replication domain consumer waits when requesting information from another replication domain consumer before giving up and assuming the information does not exist. The default is 5 seconds.

Data type	Integer
Units	Seconds
Default	5

Encryption type:

Specifies the type of encryption used before the object transfers over the network. The options include NONE, DES, TRIPLE_DES. The default is NONE. The DES and TRIPLE_DES options encrypt data sent between WebSphere Application Server processes and secure the network joining the processes.

If you specify DES or TRIPLE_DES, a key for global data replication is generated after you click **Apply** or **OK**. When you use the DES or TRIPLE_DES encryption type, click **Regenerate encryption key** at regular intervals such as monthly because periodically changing the key enhances security.

DRS partition size:

Specifies the number of groups into which a replication domain is partitioned. By default, data sent by a WebSphere Application Server process to a replication domain is transferred to all other WebSphere Application Server processes connected to that replication domain. To filter or reduce the number of destinations for the data being sent, partition the replication domain. There should be at least one server listening to every partition. If there are no servers listening on a partition, all the replicas created in that partition are lost because there is no server to cache the objects. The default partition size is 10, and the partition size should be 10 or more to enhance performance.

Partitioning the replication domain is only applicable if you are replicating data to support retrieval of an HTTP session if the process maintaining the HTTP session fails. Partitioning is not supported for sharing of cached data maintained by Web container dynamic caching. As to dynamic caching, all partitions or groups are always active and used for data replication.

When you partition a replication domain, you define the total number of groups or partitions. Use this setting to define the number of groups. Then, when you configure a specific session manager under a Web container or as part of an enterprise application or Web module, select the partition to which that session manager instance listens and from which it accepts data. To specify the groups to which an application server listens, change the settings for affected servers on a session manager page. In addition, you can set a role or runtime mode for a server. This role or mode affects whether a WebSphere Application Server process sends data to the replication domain, receives data, or does both. The default is both to receive and send data.

Data type	Integer
Default	10

Single replica:

Specifies that a single replication of data is made. Use this option only if you are using session manager with memory to memory replication. Enable this option if you are replicating data to support retrieval of an HTTP session if the process maintaining the HTTP session fails. This option restricts the recipient of the data to a single instance.

Note: Do not enable this option on a domain that is using dynamic cache replication.
This setting provides filtering beyond grouping or partitioning. Using this setting, you can choose to have data only sent to one other listening instance in the replication domain.

Default	false

Serialization method:

Specifies the object serialization method to use when replicating data. An administrative concern with replicating Java objects is locating the class definition, especially in a Java 2, Enterprise Edition (J2EE) environment where class definitions might reside only in certain web modules or enterprise applications. Object serialization methods define whether the processes receiving data also need the class definition.

The options for this setting are OBJECT and BYTES. The default is BYTES.

OBJECT instructs a replicator to write the object directly to the stream. With OBJECT, a replicator must instantiate the object on the receiving side so it must have the class definition.

BYTES instructs a replicator to break down the object into bytes and then send only the bytes across the stream. With BYTES, a replicator does not need to instantiate the object on the receiving side. The BYTES option is useful for failover, where the data is not used at the receiving side and the class definitions do not need to be stored on the receiving side. Or, the option requires that you move class definitions from the Web application class path to the system class path.

DRS pool size:

Specifies the size of the pool of resources allocated for communication with its Java Message Service (JMS) transport. You must configure this number to be the same as the DRS partition size. The default is 10.

Pooling replication resources can enhance the performance of the WebSphere internal data replication service.

DRS pool connections:

Specifies that the domain replication service should create a pool of connections with its Java Message Service (JMS) transport rather than reusing a single connection. You can pool connections when using a single replica or client server environment. You should not pool connections in a peer to peer environment.

The default is to not create a pool of connections for replication.

Replicator entry collection:

Use this page to view and manage replicator entries. Replicator entries are for use only with multi-broker replication domains. Each multi-broker replication domain consists of one or more replicator entries.

To view this administrative console page, click **Environment** > **Replication domains** > *replication_domain_name* > **Replicator entries**.

Replicator entries are only valid for multi-broker domains, which are replication domains created with a previous version of WebSphere Application Server. When you migrate your deployment manager to the current version of WebSphere Application Server, you are no longer be able to create new replicator entries in the administrative console. You can only view and modify settings for replicator entries that were created with the previous version of WebSphere Application Server.

Replicator name:

Specifies a name for the replicator entry.

Replicator entry settings:

Use this page to view and configure a replicator entry (or *replicator*). Replicators are used with multi-broker replication domains.

To view this administrative console page, click **Environment > Replication domains >** *replication_domain_name* **> Replicator entries >** *replicator_entry_name*.

Replicators communicate using Transmission Control Protocol/Internet Protocol (TCP/IP). Therefore, you must allocate an IP address and ports for replicators. Use this page to name a replicator and then to allocate an IP address and two ports (replicator and client ports) for the replicator.

Replicator name:

Specifies a name for the replicator entry.

Server:

Specifies the server for which you are defining a replicator. You can view the names of servers that do not already have replicators. You can create a maximum of one replicator on any application server.

Replicator and client host name:

Specifies the IP address, domain name service (DNS) host name with domain name suffix, or just the DNS host name, used by a client to request a Web application resource (such as a servlet, JavaServer Pages (JSP) file, or HTML page).

A replicator port and client port share the same host name.

Replicator Port:

Specifies the port for which the replicator is configured to accept messages from other replicators. The port value is used in conjunction with the host name.

The replicator port enables communication among replicators. It provides replicator port to replicator communication. The usual value specified is 7874.

Client Port:

Specifies the port for which the Web server is configured to accept client requests. The port value is used in conjunction with the host name.

The client port enables communication between an application server process and a replicator. It provides client port to application server communication. The usual value specified is 7873.

Deleting clusters

Use this task to remove a cluster and all of its cluster members.

Removing a cluster deletes the cluster and all associated cluster members. When you delete a cluster, there is no option to keep certain cluster members or applications that you have installed on any part of the cluster.

In a production environment, avoid deleting clusters that are carrying workload. You can, however, add and remove cluster members during production. When you want to remove a cluster, create a new cluster, adding new members while the old cluster is still operational. After the new cluster is working, remove the cluster members from the old cluster and then delete the cluster.

Tip: If the cluster you are removing has applications or modules mapped to it, remap the modules to another cluster, or create a new cluster and remap the modules to the new cluster, before removing the old cluster. After a cluster to which modules are mapped is deleted, the modules cannot be remapped to another cluster. Therefore, if you do not remap the modules to another cluster before deleting the old one, you must uninstall all of the modules that were mapped to the old cluster, and then reinstall them on a different cluster.

1. In the administrative console, click **Servers > Clusters**.
2. Make sure the cluster you want to remove is **stopped**. If the cluster is **started**, stop the cluster..
3. Delete the cluster. Select the cluster you want to delete, and click **Delete**.
4. Click **Save** to save your configuration changes. As part of saving your change to the configuration, select Synchronize changes with Nodes before you click **Save**.

The cluster and all of the cluster members are deleted.

Deleting specific cluster members

Use this task to remove a cluster member from an existing cluster. Removing a cluster member deletes the associated application server.

You must delete an application server to remove it from a cluster.

Note: If, in the administrative console, you select **Include cluster members in the collection** as one of your console page preferences for the Applications server page, you can use either the Application servers page or the Cluster members page to delete an application server.

To use the Cluster members page to remove an application server from a cluster:

1. In the administrative console, click **Servers > Clusters**.
2. Click the name of the cluster that contains the cluster member that you are removing from the cluster, and then click **Cluster members**.
3. Check the status of the cluster member that you are removing. If the cluster member is started, select the cluster member, and click **Stop**, and then view the Status of this cluster member again, along with any messages or logs to make sure the cluster member stops. You cannot remove a cluster member while it is running.
4. Delete the cluster member. Select the cluster member you want to delete, and click **Delete**.
5. Click **Save** to save your configuration changes. As part of saving your change to the configuration, select Synchronize changes with Nodes before you click **Save** .

The cluster member is deleted.

Configuring an application server to use the WLM even distribution of HTTP requests function

By configuring your application server to use the WLM even distribution of HTTP requests function, HTTP session objects can be evenly distributed by workload management (WLM) to the servants in your configuration. You can use this task to distribute HTTP session objects in a round-robin manner among several servants instead of the normal situation where there is a servant affinity, and HTTP session objects reside in one or two servants.

Your application server should be running on a z/OS system that is at Version 1.4 or later. Because you are distributing HTTP requests among multiple servants in this task, you should also have multiple servants enabled to use this function. See “Enabling multiple servants on z/OS” on page 379 for more information.

Use this task if your application server is experiencing problems with the default workload distribution strategy. The default workload distribution strategy uses a hot servant for running requests that create HTTP session objects. Consider configuring WebSphere Application Server and the z/OS Workload Manager to distribute your HTTP session objects in a round-robin manner in the following conditions:

- HTTP session objects in memory are used, causing dispatching affinities.
- The HTTP sessions in memory last for many hours or days.
- A large number of clients with HTTP session objects must be kept in memory.
- The loss of a session object is disruptive to the client or server.
- There is a large amount of time between requests that create HTTP sessions.

For more background about when to use this task, see “WLM even distribution of HTTP requests” on page 431.

1. In the administrative console, set the **WLMStatefulSession** property to true.
   a. Click **Servers > Applications servers**, and then select the server for which you want to use the WLM even distribution of HTTP requests function.
   b. Under Server Infrastructure, click **Administration > Administration Services**, and then, under Additional Properties, click **Custom Properties**.
   c. Click **WLMStatefulSession** and change the value in the Value field to true if it is currently set to false. The default value for this property is false for an application server and true for a deployment manager.
   d. Click **Apply** and then click **Save and Synchronize** to update your changes.
2. Set the optimal minimum and maximum number of servants for the workload. Set the minimum and maximum number of servants to handle the expected number of HTTP sessions with affinity. The minimum number of servants should be greater than one. If, for example, you expect 15,000 HTTP session objects are established in the server during the day, then you might set the minimum number of servants to some value larger than one. The minimum of servants is dependent upon the size and number of the HTTP session objects. However, the initial arrival rate of client requests establishing the affinity, the frequency of client interaction, the duration of each client interaction (CPU time and thread occupancy time), and the length of time that the HTTP session object is maintained also need to be considered when establishing the minimum value for the number of servants.
   a. To set the number of servants, click **Servers > Application servers >** *server_name* **> Server instance**.
   b. Set the minimum and maximum number of servants.
   c. Click **Save and synchronize** to apply the changes.
3. If you use a classification mapping file instead of a common workload classification document, and you specify more than one transaction class on a mapping rule for WebSphere Application Server managed round robin support, you should remove this section from your classification mapping file. You should use a common workload classification document instead of a classification mapping file because support for the classification mapping file is deprecated. However if you use a classification mapping file, and that file contains a line similar to the following:

   ```
   TransClassMap *:8080 /Dynacache1Web1/Servlet1 TCLASS1 TCLASS2 TCLASS3
   ```

   Modify this line such that it specifies only one transaction class. For example, you might change the preceding line to the following line:

   ```
   TransClassMap *:8080 /Dynacache1Web1/Servlet1 TCLASS1
   ```

   You also must update the z/OS workload manager policy to remove the extra service classes that were only necessary to get WebSphere Application Server managed round robin support. Following is an example of removing the extra service classes:

   ```
     Subsystem-Type  Xref  Notes  Options  Help
    -------------------------------------------------------------------------
                    Modify Rules for the Subsystem Type     Row 9 to 16 of 16
   ```

```
 Command ===> ___________________________________________ SCROLL ===> CSR

 Subsystem Type . : CB          Fold qualifier names?   Y  (Y or N)
 Description  . . . Component Broker requests

 Action codes:   A=After     C=Copy        M=Move     I=Insert rule
                 B=Before    D=Delete row  R=Repeat   IS=Insert Sub-rule
                                                              More ===>
           --------Qualifier--------              -------Class--------
 Action    Type       Name     Start                Service     Report
                                          DEFAULTS: AZAMS1      RBBDEFLT
  ____  1  CN         AZSR01   ___                  AZAMS1      RAZAMS1
  ____  2    TC         TCLASS1  ___                AZAMS1      RAZAMS1
  _d__  2    TC         TCLASS2  ___                AZAMS2      RAZAMS1
  _d__  2    TC         TCLASS3  ___                AZAMS3      RAZAMS1
  ____  1  CN         AZSR02   ___                  AZAMS2      RAZAMS2
  ____  1  CN         AZSR02   ___                  AZAMS3      RAZAMS3
****************************** BOTTOM OF DATA ******************************
```

4. Restart the server. The server recognizes the WLMStatefulSession property after it is restarted.

The application server uses the WLM even distribution of HTTP requests function to handle its workload instead of showing affinity to a certain servant.

See “Detecting and handling problems with runtime components” on page 282 to handle problems with server clusters and workloads.

WLM even distribution of HTTP requests

The z/OS workload management (WLM) component supports distributing incoming HTTP requests without servant affinity in a round robin manner across the servants. This functionality is intended for, but not limited to long lasting HTTP session objects that are maintained in memory, stateless session Enterprise JavaBeans (EJB), and the create method for stateful session enterprise beans. You can configure WebSphere Application Server for z/OS to use this functionality to spread HTTP requests among active servants that are currently bound to the same work queue as the inbound requests.

Background

The following diagram represents one clustered server instance. The azsr01 cluster contains the azsr01a application server instance. In the application server instance is a controller, the workload manager (WLM) queue, and the servants where applications run. The controller is the HTTP and IIOP termination point. The WLM queue controls the flow of work from the controller to one of the servants. Each of the servants contains worker threads that select work from the WLM queue.

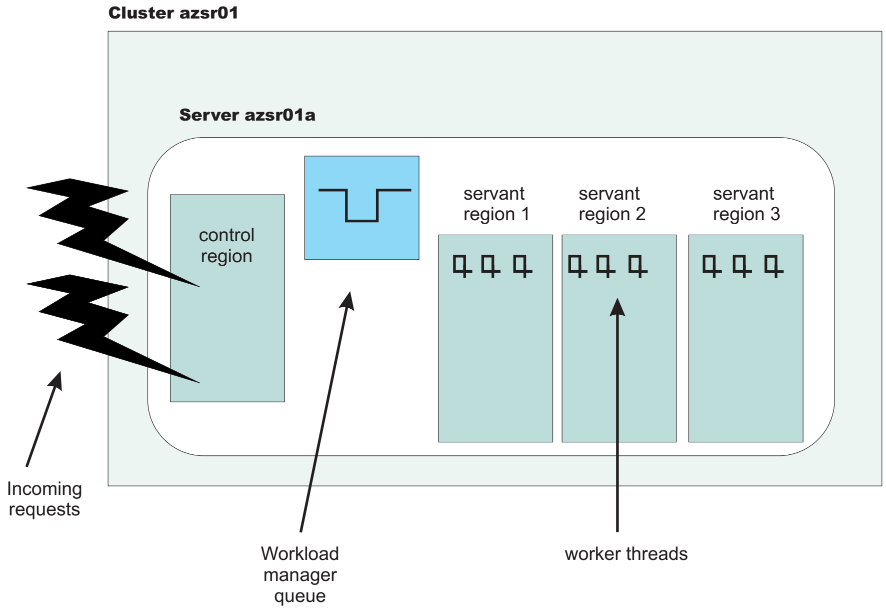
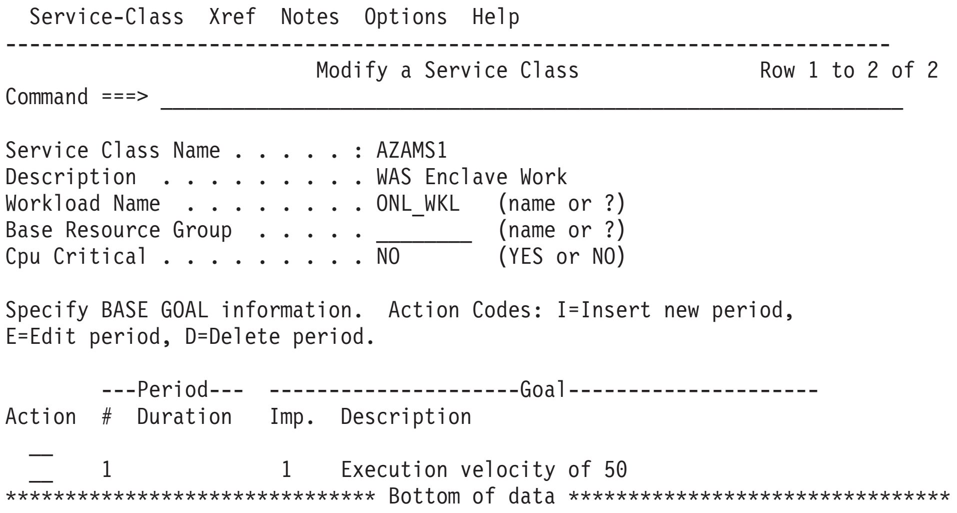


Figure 1. The contents of one clustered server instance

In the preceding diagram, the application server is configured to have the minimum and maximum number of servants set to three. For information about configuring the number of servants, see "Controlling the number of servants" on page 380.

There are WLM definitions for the application servers in this cluster. All of the requests for any application server instance in the azsr01 cluster are assigned to the same service class. The WLM classification rules assign all enclaves that are running in the azsr01a application server to the AZAMS1 service class. See the following diagrams for an example of the WLM service class definition and the classification rules.



```
  Service-Class  Xref  Notes  Options  Help
---------------------------------------------------------------------------
                          Modify a Service Class              Row 1 to 2 of 2
Command ===> _______________________________________________________________

Service Class Name . . . . . : AZAMS1
Description  . . . . . . . . . WAS Enclave Work
Workload Name  . . . . . . . . ONL_WKL    (name or ?)
Base Resource Group  . . . . . ________   (name or ?)
Cpu Critical . . . . . . . . . NO         (YES or NO)

Specify BASE GOAL information.  Action Codes: I=Insert new period,
E=Edit period, D=Delete period.

        ---Period---  ---------------------Goal---------------------
Action  #  Duration   Imp.  Description

  __
  __    1              1    Execution velocity of 50
******************************* Bottom of data ********************************
```

Figure 2. The WLM service class definition

```
  Subsystem-Type  Xref  Notes  Options  Help
--------------------------------------------------------------------------
                Modify Rules for the Subsystem Type    Row 11 to 20 of 20
Command ===> ____________________________________________ SCROLL ===> CSR

Subsystem Type . : CB          Fold qualifier names?   Y  (Y or N)
Description  . . . Component Broker requests

Action codes:   A=After     C=Copy        M=Move     I=Insert rule
                B=Before    D=Delete row  R=Repeat   IS=Insert Sub-rule
                                                             More ===>
          --------Qualifier--------                -------Class--------
Action    Type        Name      Start                Service     Report
                                          DEFAULTS: AZAMS1      RBBDEFLT
 ____   1  CN          AZSR01    ___                 AZAMS1      RAZAMS1
 ____   1  CN          AZSR02    ___                 AZAMS2      RAZAMS2
 ____   1  CN          AZSR03    ___                 AZAMS3      RAZAMS3
****************************** BOTTOM OF DATA *****************************
```

Figure 3. The WLM CB subsystem classification rules

The *hot* servant strategy

WebSphere Application Server for z/OS supports the use of HTTP session objects in memory for application servers with multiple servants. In the following diagram, two users accessed an application in the azsr01a application server instance. User 1 established an HTTP session object in servant 3. User 2 established an HTTP session object in servant 2.

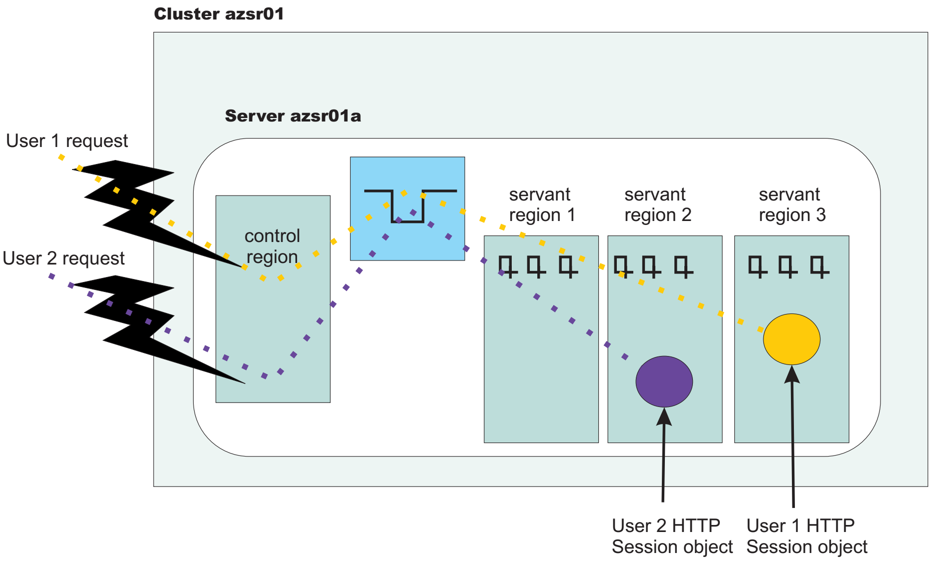


Figure 4. Users establish HTTP session objects

When a user accesses a servant region without an established HTTP session object , no servant region affinity exists. Therefore, the request can be dispatched to any servant that is available. WLM might start a new servant if all of the following conditions exist:

- The configuration allows creating new servants
- The workload manager logic determines that the system can sustain an additional servant
- Adding another servant leads to reduced queue delay and allows enclaves to be completed within the specified goal

When multiple servants are bound to the same service class, WLM attempts to dispatch the new requests to a *hot* servant. A *hot* servant has a recent request dispatched to it and has threads available. If the *hot* servant has a backlog of work, WLM dispatches the work to another servant.

Normally running this *hot* servant strategy is good because the *hot* servant likely has all its necessary pages in storage, has the just-in-time (JIT) compiled application methods saved close by, and has a cache full of data for fast data retrieval. However, this strategy presents a problem in the following situations:

- HTTP session objects in memory are used, causing dispatching affinities.
- The HTTP session objects last for many hours or days.
- A large number of clients with HTTP session objects that must be kept in memory.
- The loss of a session object is disruptive to the client or server and the amount of time between requests that create HTTP sessions is large.

In the last situation, an undesired skew in the distribution of HTTP session objects is the result. In the following diagram, most of the HTTP session objects were assigned to servant 1.

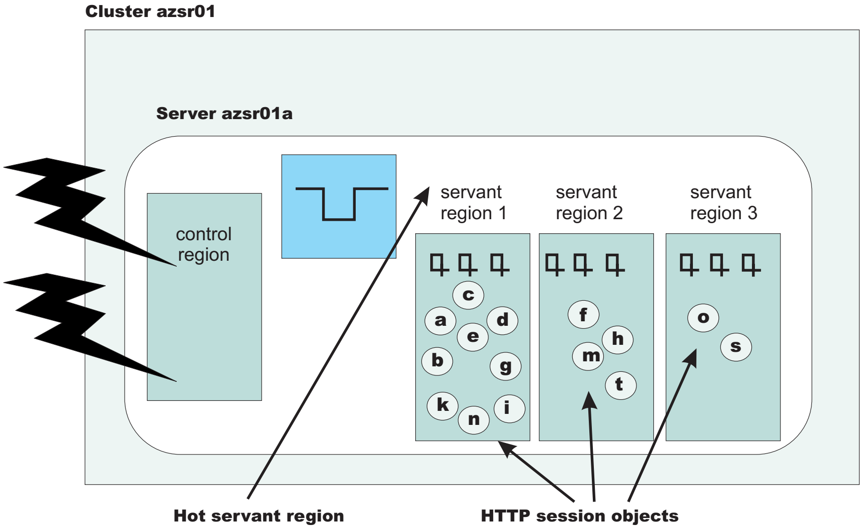


Figure 5. HTTP Session objects assigned to a hot servant

A large percentage of HTTP session objects reside in one or two servants because most of the time, there are not enough requests in the WLM queue to warrant dispatching work among many servants. This behavior can lead to the following undesirable results.

- If the application creates a large number of objects in a single servant, long garbage collection times might result.
- If all the HTTP session objects are bound to one servant, requests might be held in the queue for a long time because the work cannot be managed by WLM and cannot be dispatched in any servant.
- If all HTTP session objects reside in one or two servants, a timeout in a single servant can affect a larger number of users than if the HTTP session objects are divided equally among several servants.

Distribute incoming HTTP requests without servant affinity

If your configuration experiences one of the described situations that cause a problem with the *hot* servant strategy, you can configure your application server to support the distribution of incoming HTTP requests across servants without servant affinity. When you enable this functionality, the application server uses a round-robin distribution of HTTP requests to the servants.

In the following example, assume that the application server was configured to use the round-robin distribution of HTTP requests among the servants and multiple servants are started for the work queue requests that have the same service class assigned.

When a new HTTP request without affinity arrives on a work queue, the WLM checks to see if there is a servant that has at least one worker thread waiting for work. If there are no available worker threads in any servants, WLM queues the request until a worker thread in any of the servants becomes available. If there are available worker threads, WLM finds the servant with the smallest number of affinities. If there are servant regions with equal number of affinities, then WLM dispatches the work to the servant region with the smaller number of busy server threads.

The goal of this algorithm is for WLM to balance the incoming requests without servant affinity among waiting servants while considering changing conditions. The algorithm does not blindly assign requests to servers in a true round-robin manner. The following diagram shows the balanced distribution of HTTP session objects across servants.

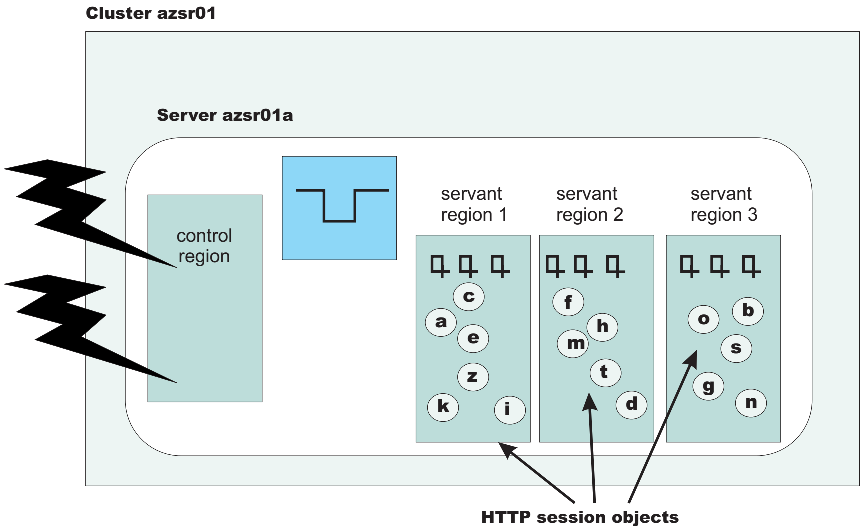


Figure 6. HTTP Session objects assigned to servants without affinity

This distribution mechanism works for all inbound requests without affinity. After the HTTP session object is created, all the client requests are directed to that servant until the HTTP session object is removed.

If you decide to enable the distribution of incoming HTTP requests without servant affinity, you might need to make some changes to your classification mapping file. If you have set up your classification mapping file to specify more than one transaction class on a mapping rule for WebSphere Application Server managed round robin support, you should remove this section from your classification mapping file.

For more information about configuring an application server to enable the distribution of incoming HTTP requests across servants without affinity, see “Configuring an application server to use the WLM even distribution of HTTP requests function” on page 429.

WLM dynamic application environment operator commands

The dynamic application environments are displayed and controlled separately from static application environments. In order to control the dynamic environments, you must set the Resource Access Facility (RACF) server class profiles to give you the proper permission to issue MVS console commands.

If you have the proper permission, you can issue the following commands from the MVS console:

Display a specific dynamic application environment

`D WLM,DYNAPPL=appl_env_name` (appl_env_name is the short cluster name)

Display all dynamic application environments

```
D WLM,DYNAPPL=*
```

Restart a specific dynamic application environment

```
V WLM,DYNAPPL=appl_env_name,RESUME
```

Quiesce a specific dynamic application environment

```
V WLM,DYNAPPL=appl_env_name,QUIESCE
```

Clustering and workload management: Resources for learning

Use the following links to find relevant supplemental information about clustering and workload management. 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

- IBM WebSphere V5.0 Performance, Scalability, and High Availability: WebSphere Handbook Series

  Note: The information presented in this Redbook is not specific to z/OS. It is written for WebSphere Application Server Network Deployment.
- IBM WebSphere V5.0 Performance, Scalability, and High Availability: WebSphere Handbook Series at http://publib-b.boulder.ibm.com/Redbooks.nsf/RedbookAbstracts/SG246198.html

  Note: The information presented in this Redbook is not specific to z/OS. It is written for WebSphere Application Server Network Deployment.
- IBM WebSphere Application Server V5.0 System Management and Configuration: WebSphere Handbook Series at http://publib-b.boulder.ibm.com/redbooks.nsf/RedbookAbstracts/sg246195.html?Open
- Workload Manager for z/OS and OS/390 at http://www-1.ibm.com/servers/eserver/zseries/zos/wlm/
- MVS Planning: Workload Management at http://publibz.boulder.ibm.com/epubs/pdf/iea2w110.pdfhttp://publibz.boulder.ibm.com/epubs/pdf/iea2w110.pdf
- MXI (MVS eXtended Information) at http://www.rocketsoftware.com/portfolio/mxi/index2.htm

Programming instructions and examples

- WebSphere Application Server education at http://www.software.ibm.com/wsdd/education/enablement/curriculum/cur_webappsrvadm.html
- Listing of all IBM WebSphere Application Server Redbooks at http://publib-b.boulder.ibm.com/Redbooks.nsf/Portals/WebSphere
- Listing of z/OS Redbooks at http://publib-b.boulder.ibm.com/redbooks.nsf/portals/S390
- Writing Optimized Java Applications for z/OS at http://publib-b.boulder.ibm.com/Redbooks.nsf/9445fa5b416f6e32852569ae006bb65f/f917e0866bcd764085256afc0066a065?OpenDocument

Chapter 9. Setting up a high availability environment

The application server infrastructure that is managed by a high availability manager includes cells and clusters. These components relate closely to core groups, high availability groups, and the policy that controls the high availability infrastructure.

Plan out how you need to set up your high availability environment to avoid the risk of a failure without failover coverage. As part of the planning process, understand how the high availability manager can assist you in controlling this type of an environment.

The high availability manager is designed to function in all of the supported WebSphere Application Server topologies. However, a high availability-managed environment must comply to the following rules:

- A cell in a high availability infrastructure is partitioned into one or more core groups. WebSphere Application Server provides a default core group as part of the high availability manager function. Additional core groups can be created using the administrative console.
- A core group cannot extend beyond the boundaries of a cell, and it cannot overlap with any other core groups.
- A cluster must be a member of only one core group. All of the individual members of that cluster must be members of the same core group.
- Individual application servers are also members of a core group.
- All running members of a core group must be able to communicate with all of the other running members of that same core group.

While administering your core groups, you might need to perform one or more of the following tasks. These tasks can be performed in any order.

1. Set up preferred coordinators.
2. Change the number of core group coordinators.
3. Configure a core group transport.
4. Configure the discovery protocol for a core group.
5. Configure the failure detection protocol for a core group.
6. Configure a core group for replication
7. Configure core group IP caching.
8. Set up IP addresses for high availability manager communications for V6.0.2 and higher.
9. Configure core group socket buffers.
10. Specify a core group when adding a node.
11. Specify a core group when creating an application server.
12. View the core groups contained in a cell.
13. View the members of a core group.
14. Create a new core group.
15. Move core group members to a different core group.
16. View high availability group information
17. Create a new policy and associate it with a high availability group.
18. Change the policy associated with a high availability group.
19. Route high availability group work to a different server.
20. Create core group access points if you create additional core groups.
21. Control application rollout and workload routing in a high availability configuration.
22. Set up a high availability sysplex.
23.

Important: After you set up your WebSphere Application Server environment to comply with all of the high availability-managed environment rules, use the default core group to control this environment. DO NOT add additional core groups unless your environment absolutely requires them. Also, do not change the default configurations unless you are doing so to solve a specific problem or situation. When you do make configuration changes, such as changing the policy for a high availability group or moving core group members between core groups in a multi-core group environment, make sure you fully understand the effect such changes will have on your entire environment.

Troubleshoot high availability environment problems. See the *Troubleshooting and support* PDF for more information.

High availability manager

WebSphere Application Server includes a high availability manager component. The services that the high availability manager provides are only available to WebSphere Application Server components.

A high availability manager provides several features that allow other WebSphere Application Server components to make themselves highly available. A high availability manager provides:

- A framework that allows singleton services to make themselves highly available. Examples of singleton services that use this framework include the transaction managers for cluster members, and the default IBM messaging provider, commonly referred to as a messaging engine.
- A mechanism that allows servers to easily exchange state data. This mechanism is commonly referred to as the bulletin board.
- A specialized framework for high speed and reliable messaging between processes. This framework is used by domain replication services when WebSphere Application Server is configured for memory-to-memory replication.

A high availability manager instance runs on every application server, proxy server, node agent and deployment manager in a cell. A cell can be divided into multiple high availability domains known as core groups. Each high availability manager instance establishes network connectivity with all other high availability manager instances in the same core group, using a specialized, dedicated, and configurable transport channel. The transport channel provides mechanisms which allow the high availability manager instance to detect when other members of the core group start, stop, or fail.

Within a core group, high availability manager instances are elected to coordinate high availability activities. An instance that is elected is known as a core group coordinator. The coordinator is highly available, such that if a process that is serving as a coordinator stops or fails, another instance is elected to assume the coordinator role, without loss of continuity.

Highly available components

A *highly available component* is a component for which a high availability group is defined on the processes where that component can run. The coordinator tracks high availability group membership, and knows on which processes each highly available component can run.

The coordinator also associates a high availability policy with each high availability group. A *high availability policy* is a set of directives that aid the coordinator in managing highly available components. For example, a directive might specify that a component runs on a specific process, if that process is available. Directives are configurable, which makes it possible for you to tailor policies to your installation.

The coordinator is notified as core group processes start, stop or fail and knows which processes are available at any given time. The coordinator uses this information, in conjunction with the high availability group and policy information, to ensure that the component keeps functioning. The coordinator uses the policy directives to determine on which process it starts and runs each component. If the chosen process fails, the coordinator restarts the component on another eligible process. This reduces the recovery time, automates failover, and eliminates the need to start a replacement process.

State data exchange

The high availability manager provides a specialized messaging mechanism that enables processes to exchange information about their current state. Each process sends or posts information related to its current state, and can register to be notified when the state of the other processes changes. The Work Load Management (WLM) component uses this mechanism to build and maintain routing table information. Routing tables built and maintained using this mechanism are highly available.

Replication

WebSphere Application Server provides a domain replication service (DRS) that is used to replicate HTTP session data, stateful EJB sessions, and dynamic cache information among cluster members. When DRS is configured for memory-to-memory replication, the transport channels defined for the high availability managers are used to pass this data among the cluster members.

When to use a high availability manager

A high availability manager consumes valuable system resources, such as CPU cycles, heap memory, and sockets. These resources are consumed both by the high availability manager and by WebSphere Application Server components that use the services that the high availability manager provides. The amount of resources that both the high availability manager and these WebSphere Application Server components consume increases exponentially as the size of a core group increases. For large core groups, the amount of resources that the high availability manager consumes can become significant. If the services that the high availability manager provides are not used, then disabling the high availability manager frees these resources.

The capability to disable the high availability manager is most useful for large topologies where none of the high availability manager provided services are used. In certain topologies, only some of the processes use the services that the high availability manager provides. In these topologies, you can disable the high availability manager on a per-process basis, which optimizes the amount of resources that the high availability manager uses.

Do not disable the high availability manager on administrative processes, such as node agents and the deployment manager, unless the high availability manager is disabled on all application server processes in that core group.

Some of the services that the high availability manager provides are cluster based. Therefore, because cluster members must be homogeneous, if you disable the high availability manager on one member of a cluster, you must disable it on all of the other members of that cluster.

When determining if you must leave the high availability manager enabled on a given application server process, consider if the process requires any of the following high availability manager services:

- Memory-to-memory replication
- Singleton failover
- Workload management routing
- On-demand configuration routing

Memory-to-memory replication

Memory-to-memory replication is a cluster-based service that you configure or enable at the application server level. If memory-to-memory replication is enabled on any cluster member, then the high availability manager must be enabled on all of the members of that cluster. Memory-to-memory replication is automatically enabled if:

- Memory-to-memory replication is enabled for Web container HTTP sessions. See the *Developing and deploying applications* PDF for more information.
- Cache replication is enabled for the dynamic cache service. See the *Developing and deploying applications* PDF for more information.
- EJB stateful session bean failover is enabled for an application server.See the *Developing and deploying applications* PDF for more information.

Singleton failover

Singleton failover is a cluster-based service. The high availability manager must be enabled on all members of a cluster if one or more instances of the default Java Message Service (JMS) provider are configured to run in the cluster. The default JMS provider is the messaging engine that is provided with WebSphere Application Server.

Workload management routing

Workload management (WLM) propagates the following classes or types of routing information:
- Routing information for enterprise bean Internet Inter-ORB Protocol (IIOP) traffic.
- Routing information for the default IBM Java Messaging Service (JMS) provider, which is also referred to as the messaging engine.

WLM uses the high availability manager to both propagate the routing information and make it highly available. Although WLM routing information normally applies to clustered resources, it can also apply to non-clustered resources, such as standalone messaging engines. Under normal circumstances, you must leave the high availability manager enabled on any application server that produces or consumes either IIOP or messaging engine routing information. For example if:
- The routing information producer is an enterprise bean application that resides in cluster 1.
- The routing information consumer is a servlet that resides in cluster 2.

When the servlet in cluster 2 calls the enterprise bean application in cluster 1, the high availability manager must be enabled on all servers in both clusters.

Workload management provides an option to statically build and export route tables to the file system. Use this option to eliminate the dependency on the high availability manager. See “Enabling static routing for a cluster” on page 408 for more information about the Export route table option.

On-demand configuration routing

In a Network Deployment system, the on-demand configuration is used for both proxy server and Web services routing. The high availability manager must be enabled on all processes to which the proxy server routes work, and on all processes that are running Web services.

Best practices

Although not required, core groups are usually homogeneous. If you have a large installation and you need to set up a mix of processes that do and do not use the high availability manager, you can:
- Create a new core group and move all application servers on which the high availability manager is disabled to this core group. A core group that contains only application servers on which the high availability manager is disabled does not have to contain an administrative process and have no restrictions on how large such a core group can become.
- Leave the remaining applications servers that require high availability manager services in the default core group. You can create additional core groups for some of the application servers that require the high availability manager if the default core group becomes too large.

Core groups

A core group is a high availability domain that consists of a set of processes in the same cell that can directly establish high availability relationships. Highly available components can only fail over to another process in the same core group and replication can occur only between members of the same core group.

A cell must contain at least one core group, although multiple core groups are supported. Each core group contains a core group coordinator to manage its high availability relationships, and a set of high availability policies that are used to manage the highly available components within that core group.

Core group members

Every deployment manager, node agent, application server, and proxy server is a member of a core group. When a process is created it is automatically added to a core group. The core group membership is stored in a WebSphere Application Server configuration document. You can move processes from one core group to another. The following rules govern the core group membership:

- Every process is a member of exactly one core group.
- All members of a cluster must be members of the same core group.
- Each core group must contain at least one node agent or the deployment manager.

A core group member has a well-defined life cycle. When the first core group member starts, the transport that is dedicated to that core group automatically starts. The Discovery Protocol, View Synchrony Protocol, and Failure Detection Protocol for that core group member also start and run for the entire lifetime of the core group member:

- The Discovery Protocol is responsible for discovering when other core group processes start, and for opening network connections to these other members.
- The View Synchrony Protocol is responsible for establishing reliable messaging with other core group members after the connections are opened.
- The Failure Detection Protocol is responsible for detecting when other core group members stop or become unreachable because of a network partition.

Core group coordinator

The core group coordinator is responsible for coordinating high availability activities between the core group members for which View Synchrony Protocol is established.

Core group transport

Network communication between all the members of a core group is essential. The network environment must consist of a fast local area network (LAN) with full Internet Protocol (IP) visibility and bidirectional communication between all core group members. Each core group member must be able to receive communications from any of the other core group members.

Multiple core groups

A cell, by default, contains a single core group, called DefaultCoreGroup. All processes in the cell are initially members of this core group. A single core group is usually sufficient. However, some topologies or special circumstances require multiple core groups. There are also topologies that do not require multiple core groups but having them is a good practice. For example, you might want to define multiple core groups if :

- There are one or more firewalls within a cell. A core group can not contain members from multiple firewall protection domains.
- A large number of processes in the cell and the core group protocols, such as the View Synchrony Protocol, consume correspondingly large amounts of resources such as CPU.

- Core group protocols, such as the Failure Detection Protocol, need tuning or configuring to use values that work best with smaller numbers of core group members.

If members of different core groups need to share WLM routing information, use the core group bridge service to connect these core groups. The core group bridge service uses access point groups to connect the core groups. A core group access point defines a set of bridge interfaces that resolve to IP addresses and ports. The core group bridge service uses this set of bridge interfaces to enable members of one core group to communicate with members of another core group.

Core group coordinator

Every core group has a coordinator that manages high availability activities between the core group members. The coordinator manages the failover of highly available singleton services and distributes live server state data to interested core group members. The coordinator uses some CPU and memory (JVM heap) resources to perform these tasks. In some configurations, the amount of resources that the coordinator uses might be large.

The coordinator workload can be divided over multiple coordinator instances. Each instance runs on a different core group member and is assigned a portion of the overall coordination workload. Dividing the workload across multiple coordinator instances enables you to share the associated resource costs across machines. The coordinator function remains highly available, regardless of how its workload is divided or assigned to core group members.

Coordinator election

When a core group member starts or stops, the View Synchrony Protocol installs a new view. The view consists of the core group members that are connected and cooperating. Whenever a new view is installed, it might be necessary to redivide the coordinator workload among the core group members. For example, a core group member that is hosting a coordinator instance might fail and the high availability manager must elect a replacement coordinator.

Informational messages, similar to the following message, are logged in the SystemOut.log file when a particular core group member is elected as a coordinator:

```
HMGR0206I: The coordinator is an active coordinator for core group DefaultCoreGroup
```

Messages, similar to the following message, are logged if a core group member is no longer an elected coordinator:

```
HMGR0207I: The coordinator was previously an active coordinator for core group
           DefaultCoreGroup but has lost leadership.
```

Important: Remember that coordinator election occurs whenever the view changes. Electing a new coordinator uses a lot of resources because this process causes increased network traffic and CPU consumption. Specifying a preferred coordinator server, whenever practical, helps eliminate the need to make frequent coordinator changes.

Multiple coordinators

The WebSphere core group configuration data contains a field in which users can specify the number of coordinators. The default value for this field is 1. This default value is sufficient for most installations and applications. Use multiple coordinators when the core group member that is selected as the coordinator uses noticeably more memory or CPU than similar core group members. In addition, some software products that heavily use the high availability framework instruct you to increase the number of coordinators.

Preferred servers

When you configure a core group, you can specify the core group members the high availability manager should use as coordinators, if they are available. Preferred coordinator servers should be core group processes that are cycled as infrequently as possible. The preferred coordinator servers should also be hosted on machines with excess capacity.

Specifying preferred coordinator servers is a good practice. When coordinators are elected during a view change, the high availability manager checks for a list of preferred servers. If there is a list, the high availability manager selects a server from that list as the coordinator. If there is no list, the high availability manager selects the view member with the lexically lowest name as the coordinator, which incurs some overhead if it causes the coordinator to move.

Core group administration considerations

Core group configuration information is stored in a CoreGroup configuration object that is backed by a `coregroup.xml` document. Process-specific configuration information for each core group member is stored in a HAManagerService configuration object that is backed by a `hamanagerservice.xml` document.

The `coregroup.xml` document is a cell-scoped document. The master copy of this document is stored in the configuration repository for the deployment manager. A copy of this document is shadowed to every node in the cell. The `coregroup.xml` document includes of the following configuration information:

- The list of core group members
- The high availability policies for the core group
- The core group coordinator configuration information
- The core group transport configuration information

The core group member process-specific configuration information stored in the `hamanagerservice.xml` document includes:

- Whether the high availability manager is enabled.
- The transport buffer size.
- The name of the core group to which the member belongs.
- How frequently the high availability manager checks the health of highly available singletons running on the member, if a length of time is in affect for this function.

Core group configuration document

The master copy of the core group configuration document is directly modified when direct attributes, such as the coordinator configuration, are modified. The master copy of the core group configuration document is implicitly modified when a server is created or deleted, or a node is added or removed. In either case, the list of core group members is updated to reflect which processes are added or removed.

The set of core group members for which the View Synchrony Protocol is established is commonly referred to as a *view*. Whenever a view is installed, one of the core group members is elected to send its current configuration to all other members of the view. This processing ensures that all members of the view are running with a consistent core group configuration. This processing also means that inconsistencies in a high availability policy or coordinator configuration are tolerated. However, inconsistencies in the list of core group members or the core group transport are not tolerated.

Before you modify a list of core group members, remember that all core groups must contain at least one administrative process. In a situation where the configuration document is synchronized to all of the nodes in the cell, and you have multiple core group processes running, the running core group administrative processes are notified whenever the configuration document is modified. The high availability manager selects one of the administrative processes to reread the configuration and distribute the updated configuration to all of the other core group members in the same view. These changes are then dynamically picked up by all of these other core group members. If the core group does not contain at least one administrative process that is running when a configuration change is made, the updated configuration is not properly passed on to the core group members.

If you modify a list of core group members, do not start a member of that core group until you are sure that the change is fully synchronized to all nodes in the cell. If a node agent is down when the configuration change is made, you must manually synchronize the configuration change before any processes are started on that node. If you do not manually synchronize the change, the process that is starting cannot establish the View Synchrony Protocol with the other core group members because when a core group member starts, it reads the core group configuration information from the repository on the local node. It then opens connections to other core group members and attempts to establish the View Synchrony Protocol with them. If the local copy of the `coregroup.xml` document is not synchronized with the master core group configuration document, problems occur. For example, if the running processes dynamically reloaded the updated configuration, the configuration for the process that just started is out of sync with the configurations of the other core group members. If the update changed the list of core group members, the list is now inconsistent across the nodes in the cell, and any attempt to establish view synchrony fails because of these inconsistent member lists. When this condition is detected, an error message similar to the following message is logged:

```
DCSV8022I: DCS Stack {0} at Member {1}: Inconsistency of configured defined set
with that of another member. Inconsistent member is {2}. The list of members only
in the local defined set is {3}, whereas the list of members only in the defined
set at the inconsistent member is {4}.
```

When a process detects an inconsistent core group membership condition, the process attempts to reread the core group configuration several times. It is possible that the configuration document is in the process of being synchronized to the node. In such a case, rereading the configuration document can resolve the inconsistency. However, if the process can not resolve the inconsistency after trying to reread the configuration several times, the process stops trying to resolve the inconsistency. To recover from this situation, you must resynchronize the configuration and restart the process.

Core group process-specific configuration document

Unlike the cell-scoped core group configuration information that is contained in the `coregroup.xml` document, the process-specific configuration information for each core group member that is contained in the `hamanagerservice.xml` document cannot be dynamically reloaded. You must restart a process before core group process-specific configuration changes go into affect.

Core group scaling considerations

The amount of system resources, such as CPU and memory, that the high availability manager consumes does not increase linearly as the size of a core group increases. For example, the View Synchrony Protocol that the high availability manager uses requires a large amount of these resources to maintain a tight coupling over the core group members. Therefore, the amount of resources that a large core group consumes might become significant.

View Synchrony Protocol resource requirements can vary considerably for different core groups of the same size. The amount of resources that the View Synchrony Protocol uses for a core group is determined by:

- The number of applications that are running.
- The type of applications that are running.
- The high availability manager services that are used.

When setting up core group scalability, you must ensure that:

- All of the processes within the cell are distributed properly into core groups of appropriate sizes. Properly distributing these processes limits the amount of resources that the View Synchrony Protocol consumes.

- All of the processes within a given core group are properly configured to support the high availability services that are used within the core group.

Consider implementing one or more of the following scalability techniques to scale the high availability manager in large cells, even if your system is operating properly. The two most basic techniques are:

- Disabling the high availability manager if it is not required.
- Distributing the processes over a number of core groups and using a core group bridge to connect the core groups as required.

Adjusting the size of a core group

Core group size directly effects three aspects of high availability manager processing that impact resource usage:

- The first and most significant aspect is the establishment of the View Synchrony Protocol over a set of active core group members. This activity is commonly referred to as a *view change*.
- The second aspect is the regularly scheduled discovery and failure detection tasks that high availability manager runs in the background.
- The third aspect is the resource usage that results when other WebSphere Application Server components use high availability manager-provided services.

View Changes

The View Synchrony Protocol creates a new view whenever it detects that there is a change in core group members that are active. A view change typically occurs whenever a core group member starts or stops. When a core group member starts, it opens a connection to all of the other running core group members. When a core group member stops, other core group members detect that their open connections to the stopped member are closed. In either case, the View Synchrony Protocol needs to account for this change. In the case of a newly started member, the View Synchrony Protocol must establish a view that includes the new member. In the case of a stopped member, the View Synchrony Protocol must establish new view for the surviving core group members that excludes the stopped member.

Establishing a new view is an important activity but uses a lot of system resources, especially for large core groups.

- Each running core group member must communicate its current state to other core group members, including information about the messages it has sent or received in the current view.
- All messages sent in a given view must be received and acknowledged by all recipients before a new view can be installed. Under normal operating conditions, receipt of these messages is acknowledged slowly. Completing messages at a view change boundary in a timely fashion requires aggressive acknowledgement and retransmission.
- All core group members must transmit data regarding their current state, such as the set of other core group members to which they can actively communicate.

As the number of active members grows, installing a new view requires a larger, temporary nonlinear increase in high availability manager CPU usage. It is significantly more expensive to add or remove a single member when 50 other core group members exist, than it is to add or remove a member when 20 other members exist.

Installing a new view also triggers state changes in WebSphere Application Server components that use the high availability manager. For example, routing tables might need to be updated to reflect the started or stopped member, or a singleton service might need to be restarted on a new member.

The end result is that installing a new view results in a significant, transient spike in CPU usage. If core group sizes become too large, degenerate network timing conditions occur at the view change boundary. These conditions usually result in a failure during an attempt to install a new view. Recovery from such a failure is also CPU intensive. When insufficient CPU is available, or paging occurs, failures can quickly multiply.

Background tasks

The high availability manager periodically runs a number of background tasks, such as checking the health of highly available singleton services that it is managing. Most of these background tasks consume trivial amounts of CPU. The exceptions are the regularly scheduled discovery and Failure Detection Protocols.

The Discovery Protocol attempts to establish communications among core group members that are not currently connected, including processes that are not running. For a given core group that contains N core group members, of which M are currently running, each discovery period results in roughly M x (N – M) discovery messages. Therefore, creating a large number of processes that never start adversely affects the Discovery Protocol CPU usage.

Similarly, when the Failure Detection Protocol runs, each core group member sends heartbeats to all of its established connections to other core group members. For M active members, M x (M-1) heartbeat messages are sent. If aggressive failure detection is required, the size of the core group can adversely affect the amount of CPU usage that heartbeating between core group members consumes.

Smaller core groups positively affect the amount of CPU usage these two protocols consume. For example, if a core group contains 100 active members, 9900 heartbeat messages are sent during every failure detection period. Splitting the 100 member core group into five smaller core groups of 20 members reduces this number of message to 1900, which is a significant reduction.

External usage

Other WebSphere Application Server components, such as work load management (WLM), and on demand configuration, use high availability manager-provided services, such as live server state exchange, to maintain routing information. The amount of CPU usage that these components consume is linked to core group size. For example, the usage of the live server state exchange to build highly available routing information is linked to the size of the core group.

Distributing processes among multiple core groups

You can use two basic techniques to minimize the amount of resources that the view synchrony and related protocols consume:

- You can disable the high availability manager on processes where the services that the high availability manager provides are not used.
- You can keep core group sizes small.

The key to limiting the high availability manager CPU usage is to limit the size of the core group. Multiple small core groups are much better than one large core group. If you have large cells, create multiple core groups.

The hardware on which you are running WebSphere Application Server is also a factor in determining the core group size that is appropriate for your environment.

The current recommendation is to limit core group sizes to 50 members or so. Split large core groups into multiple, smaller core groups. If the resulting core groups need to share routing information, you can use core group bridges to bridge the core groups together.

Adjusting individual core groups based on the application mix and services used

You might need to further adjust Individual core groups based on the application mix and the high availability services that the core group members use.

- Adjust how frequently the Discovery Protocol and the Failure Detection Protocol run if the default settings are not appropriate.
- Configure the core group coordinator to run on a specific process or set of processes.
- Partition the coordinator across multiple instances if the consumption of resources by the coordinator process is noticeable.
- Configure the amount of memory that is available to the distribution and consistency services (DCS) and removable media manager (RMM) components for sending network messages when congestion is detected. Congestion can occur under some conditions, even though memory-to-memory replication is not used.

Adjusting ephemeral port ranges

The number of sockets that a core group uses is usually not a major concern. Each core group member must establish a connection with every other member of that core group. Therefore, the number of connections grows exponentially (n-squared) because each connection requires two sockets, one on each end of the connection. Because multiple machines are typically involved, normally you do not have to be concerned about the number of sockets that a core group uses. However, if you have an abnormally large number of core group members that are running on a single machine, you might have to adjust the operating system parameters that are related to ephemeral port ranges. Most operating systems have different default behavior for ephemeral port ranges.

Core group transports

Core group members communicate with each other over a specialized and dedicated network transport. Multiple transport implementations are supported, each with its own set of advantages and disadvantages.

You can use one of the following types of transports to set up communications between core group members:

- Channel framework transport, which is the default
- Unicast transport
- Multicast transport

All of these transport options require TCP/IP connections for communication between core group members. The high availability discovery protocol ensures that these connections are opened, but the selected transport opens the connections. The discovery protocol also ensures, for each core group member, that connections are opened to all other members of that core group, thereby ensuring that all running core group members are fully connected through TCP/IP.

Each core group member has a configured endpoint known as the DCS_UNICAST_ADDRESS. This endpoint contains the host and port information that indicates where the core group member is listening for TCP/IP connections.

Channel framework transport

The channel framework transport is the most flexible and scalable of the transport options and is the best option for most topologies. The channel framework transport provides the most security options for core group connections. However, the advanced features of the channel framework transport come at the cost of slightly lower performance. This performance impact is a concern only in the most demanding topologies where replication throughput is a primary objective.

The channel framework transport is the default transport type for core group member connections. If you use a channel framework transport, all communication occurs directly over the core group TCP/IP connections. The work of opening connections and sending or receiving data is delegated to the channel framework transport and the associated channel implementations.

For example, to use a channel framework transport for distribution and consistency services (DCS) messages, you must configure either a DCS transport chain, which is the default, or a DCS_SECURE transport chain, in addition to the DCS_UNICAST_ADDRESS endpoint. A DCS transport chain uses a TCP channel for network communication. A DCS_SECURE transport chain includes both a TCP channel and an secure sockets layer (SSL) channel to add support for encrypted communication.

The channel framework function provides a common model for connection management, and contains support for implementing various channels that you can combine into a transport chain. The ability to combine various types of channels makes it possible to create custom transport chains. The channel framework function also supports port sharing, which provides additional flexibility and the potential for future customization.

If you use a channel framework transport, two different mechanisms are available to secure the core group network connections:

1. A lightweight third-party authentication (LTPA) token is used to authenticate all incoming connection requests if WebSphere Application Server administrative security is enabled.
2. Secure Sockets Layer (SSL) is used to encrypt all communications if the SSL version of a transport chain is selected.

You can use these mechanisms separately or combine them for the highest level of security possible.

Unicast transport

The communication mechanism of the unicast transport is similar to that of the channel framework transport. All communications occur over core group TCP/IP connections. The major difference is that a standard network connection, instead of a transport chain, is established and used to perform the communication between core group members.

Because a unicast transport does not go through the channel framework, this transport is somewhat faster than the channel framework transport. This performance improvement might be useful in topologies that make extensive use of memory-to-memory replication, and where the throughput that is obtained using a channel framework transport is not adequate.

Internal benchmarks show a gain in throughput using a unicast transport rather than a channel framework channel. However, this performance gain is achieved at the cost of using additional ephemeral ports.

A unicast transport has fewer security options than the channel framework transport. As with the channel framework transport, if WebSphere administrative security is enabled, an LTPA token is used to authenticate all incoming connection requests. However, no SSL encryption option is available for a unicast transport.

A unicast transport provides optimum performance with basic security. You trade the flexibility that a channel framework transport provides for improved performance. You are no longer able to do things like configure for SSL encrypted communication, or create a custom transport chain. However, in a typical application server environment, the unicast transport provides the best performance for functions like memory-to-memory session replication.

Because a unicast transport uses more ephemeral ports than a channel framework transport, it might not scale as well as a channel framework transport does as the core group size increases.

Multicast transport

The multicast transport uses IP multicast and broadcasts information to all of the other processes in the core group. Although IP multicast uses a user datagram protocol (UDP), failure detection protocol still requires TCP/IP connections to be set up between core group members.

The multicast transport requires the following additional configuration parameters:

- Multicast port, for which the default value is 23445
- Multicast IP range start, for which the default value is 239.0.0.0
- Multicast IP range end, for which the default value is 239.255.255.255

A multicast transport also provides a high level of performance. However, because a multicast transport broadcasts information to all of the members of a core group, it is best suited for situations where many core group members need the same information. If you use a multicast transport when only a few members need the same information, you risk saturating the network with unnecessary data packets.

The multicast transport is the least secure of the available core group transport types because it broadcasts information out to all members of the networks. There are no restrictions as to who could potentially receive the data or try to send data to the connection. By default, the Time to Live (TTL) field in the IP header is 1. Therefore, multicast data is restricted to the subnet on which it is sent. If all of the systems are protected and located on the same subnet, then additional levels of security might not be required.

The multicast transport is optimum when many members in the core group need to have data replicated across them. The multicast transport typically requires that all members of the core group be located on a single subnet.

Core group Discovery Protocol

When a core group member starts, no connections to other core group members exist. The task that runs the Discovery Protocol for this core group member starts as part of the core group members startup procedure. The Discovery Protocol establishes network connectivity with the other members of the core group. This task runs at regularly scheduled intervals as long as the core group member is active.

The Discovery Protocol retrieves the list of core group members and the associated network information from the WebSphere Application Server configuration settings. The Discovery Protocol then attempts to open network connections to all of the other core group members. At periodic intervals, the Discovery Protocol recalculates the set of unconnected members and attempts to open connections to those members.

When a connection is made to another core group member, the Discovery Protocol notifies the View Synchrony Protocol, and logs this event as an informational message, similar to the following message, in the SystemOut.log file.

```
DCSV1032I: DCS Stack DefaultCoreGroup at Member MyCell\anzio\nodeagent:
Connected a defined member MyCell\anzioCellManager\dmgr.
```

Connections can fail at any time for a variety of reasons. The Failure Detection Protocol detects connection failures and notifies the Discovery Protocol. The Discovery Protocol then attempts to open a new network connection to that member at the next scheduled interval.

The interval at which the Discovery Protocol runs is configurable. For Versions 6.0 and 6.0.1, the default is 15 seconds. For Version 6.0.2 and higher, the default is 30 seconds. You can use the IBM_CS_UNICAST_DISCOVERY_INTERVAL_SECS core group custom property to change this setting.

The amount of CPU cycles that the Discovery Protocol task consumes is proportional to the number of core group members that are stopped or unreachable. The CPU cycles that the Discovery Protocol task consumes is negligible at the default settings.

Core group Failure Detection Protocol

When a core group member starts, a task running the Failure Detection Protocol also starts. This task runs as long as the member is active. The Failure Detection Protocol monitors the core group network connections that the Discovery Protocol establishes. When the Failure Detection Protocol detects a failed network connection, it reports the failure to the View Synchrony Protocol and the Discovery Protocol. The

View Synchrony Protocol adjusts the view to exclude the failed member. The Discovery Protocol attempts to reestablish a network connection with the failed member.

The Failure Detection Protocol uses two distinct mechanisms to find failed members:

- It looks for connections that closed because the underlying socket was closed.
- It listens for active heartbeats from the core group members.

Sockets closing

When a core group member normally stops in response to an administration command, the core group transport for that member also stops, and the socket that is associated with the transport closes. If a core group member terminates abnormally, the underlying operating system normally closes the sockets that the process opened and the socket associated with the core group transport. is closed.

For either type of termination, core group members that have an open connection to the terminated member are notified that the connection is no longer usable. The core group member that receives the socket closed notification considers the terminated member a failed member.

When a failed member is detected because of the socket closing mechanism, one or more of the following messages are logged in the SystemOut.log file for the surviving members:

```
DCSV1113W: DCS Stack DefaultCoreGroup at Member anzioCell01\anzioCellManager01\dmgr:
Suspected another member because the outgoing connection to the other member was closed.
Suspected member is anzioCell01\nettuno\ServerB. DCS logical channel is View|Ptp.

DCSV1111W: DCS Stack DefaultCoreGroup at Member anzioCell01\anzioCellManager01\dmgr:
Suspected another member because the outgoing connection from the other member was closed.
Suspected members is anzioCell01\nettuno\ServerB. DCS logical channel is Connected|Ptp.
```

The closed socket mechanism is the way that failed members are typically discovered. TCP settings in the underlying operating system, such as FIN_WAIT, affect how quickly socket closing events are received.

Active heart beating

The active heart beating mechanism is analogous to the TCP the keep alive function. At regularly scheduled intervals, each core group member sends a ping packet on every open core group connection. If the packet is acknowledged, all is assumed to be all right. If no response is received from a given member for a certain number of consecutive pings, the member is marked as failed. When a member is marked as failed, the following message is logged:

```
DCSV1112W: DCS Stack DefaultCoreGroup at Member anzioCell01\anzioCellManager01\dmgr:
Suspected member anzioCell01\nettuno\ServerB because of heartbeat timeout.
Configured Timeout is 180000 milliseconds. DCS logical channel is Connected|Ptp.
```

Active heartbeats are most useful for detecting core group members that are unreachable because the network is stopped. Active heartbeats consume some CPU usage. The amount of CPU usage that is consumed is proportional to the number of active members in the core group. The default configuration for active heartbeats is a balance of CPU usage and timely failed member detection. You can use the following core group custom properties to change the settings for active heartbeats:

- IBM_CS_FD_PERIOD_SECS, which specifies the time interval, in seconds, between consecutive heartbeats. The default value for this property is 30 seconds.
- IBM_CS_FD_CONSECUTIVE_MISSED, which specifies the consecutive number of heartbeats that must be missed before the core group member is considered failed. The default value for this property is 6.

Core group View Synchrony Protocol

The View Synchrony Protocol is established over the set of core group members that can communicate with each other. This protocol provides guaranteed, in-order message delivery for message streams that involve one sender and potentially multiple receivers. This guarantee is similar to the guarantees that TCP/IP provides for point-to-point message streams.

The set of core group members for which the View Synchrony Protocol is established is commonly referred to as a *view*. Views are unique in time and space. The act of adding or removing members from the view is called a *view change*. A view change is an important and relatively expensive synchronization point. It is also the point where synchronization, consistency and network issues are detected.

The View Synchrony Protocol is transparent to both components using the high availability manager framework and WebSphere Application Server administrators. However, disruptions in the View Synchrony Protocol might become visible, most notably when a boundary condition known as a view change occurs.

View changes

When a core group member starts, the core group transport and the associated Discovery Protocol, Failure Detection Protocol, and View Synchrony Protocol also start. The View Synchrony Protocol establishes an initial view that contains only the local member. The View Synchrony Protocol is notified when the Discovery Protocol establishes connections with other core group members. The view synchrony layer of the newly connected members then exchange state information. This information is used to determine if a new view can be formed. For example, if a newly started member discovers an existing view, it negotiates with the members of the existing view to establish a new view.

When a member of an established view stops or fails, the Failure Detection Protocol on the surviving view members detects the failure and notifies the View Synchrony Protocol. The surviving members then establish a new view that excludes the failed member.

Before a new view is established, activities that are related to the current view must be completed. All messages that are sent in the current view must be received and acknowledged by all intended recipients that are still alive. The current members must exchange a non-trivial amount of state information regarding messages sent and received. These members then perform the activities that are needed to complete the pending message activity, which might include the retransmission of messages that seem to be lost.

Installing a new view might result in significant, temporary spikes in the amount of CPU consumed and the network bandwidth used.

View change messages

A view change is a complex multipart procedure and a number of messages are logged every time a view is changed. These messages indicate the stage of view change processing that is complete or is currently running.

For example, the following message indicates that a set of core group members agreed to establish a new view and initiated the view change procedure:

```
DCSV8054I: DCS Stack DefaultCoreGroup at Member
anzioCell01\anzioCellManager01\dmgr: View change in process.
```

The following message indicates that all messages sent in the current view are completed and acknowledged:

```
DCSV2004I: DCS Stack DefaultCoreGroup at Member
anzioCell01\anzioCellManager01\dmgr: The synchronization procedure completed
successfully. The View Identifier is (2:0.anzioCell01\anzioCellManager01\dmgr).
The internal details are [0].
```

The following messages indicate that the view change completed successfully. They also specify the name or identifier for the new view, and the number of core group members in that view:

```
HMGR0218I: A new core group view has been installed. The core group is
DefaultCoreGroup. The view identifier is (3:0.anzioCell01\anzioCellManager01\dmgr).
The number of members in the new view is 2.

DCSV1033I: DCS Stack DefaultCoreGroup at Member
anzioCell01\anzioCellManager01\dmgr: Confirmed all new view members in view
identifier (3:0.anzioCell01\anzioCellManager01\dmgr). View channel type is View|Ptp.
```

The following message provides extended status regarding the state of connections and view synchrony:

```
DCSV8050I: DCS Stack DefaultCoreGroup at Member
anzioCell01\anzioCellManager01\dmgr: New view installed, identifier
(3:0.anzioCell01\anzioCellManager01\dmgr), view size is 2 (AV=2, CD=2, CN=2, DF=6)
```

In this message:

- AV is the number of core group members in the view.
- CN is the number of core group members to which this member has open connections. Normally this number is the same as the number that is specified for AV.
- CD is the number of core group members to which this member has open connections minus the number of bad members. A bad member is one that is connected to this member, but cannot currently establish a view with this member.
- DF is the number of members defined in the core group.

High availability groups

High availability groups are part of the high availability manager framework. A high availability group provides the mechanism for building a highly available component and enables the component to run in one of several different processes. A high availability group cannot extend beyond the boundaries of a core group.

A high availability group is associated with a specific component. The members of the group are the set of processes where it is possible to run that component. Therefore, a WebSphere Application Server administrator cannot directly configure or define a high availability group, and its associated set of members. Instead high availability groups are created dynamically at the request of the components that need to provide a highly available function.

Scope

A high availability group cannot extend beyond the boundaries of a core group. Therefore, a highly available component cannot fail over from a server process that is defined in one core group to a server process that is defined in a different core group.

Life cycle

Because high availability groups are dynamically created, a WebSphere Application Server administrator has no direct control over when they are created or destroyed. A high availability group is created when component code that runs in a given process calls the high availability manager framework to join a group. The calling component must provide the name of the high availability group for the high availability manager framework to join.

If a high availability group with this name does not currently exist, the high availability manager creates one, and makes this member the first member of the newly created group. If the high availability group already exists, this member is added to the set of high availability group members.

Because several different components might use the high availability manager framework, it is possible to have several different high availability groups across the same set of processes. However, each high availability group always has a unique group name.

A high availability group ceases to exist when all of the group members leave the group, which typically occurs when all of the processes that host members of a given high availability group stop.

Group name

Every high availability group has a unique name. Because any component can create a high availability group for that component to use, it is the high availability group name that ties a given component to a particular high availability group. A high availability group name is not a simple string; this name is a set of name-value pairs that the creating component specifies. A high availability group name can look like the following example:

```
Company=IBM,ComponentName=TM,policy=DefaultNoQuorumOneOfNPolicy
```

A component can specify any number of name-value pairs to create a unique name for their high availability group.

Member state

Each member of a high availability group is either idle, active or disabled. Typically, a high availability group member will be either idle or active. A member that is idle is not assigned any work, but is available as a backup if a member that is active fails. A member that is active is designated as the member to handle the component workload.

If a member is disabled, it cannot participate in the high availability group. A disabled member is not assigned any work, and is not available as a backup if an active member fails. An administrator might disable a member if they plan to remove, delete, or cycle power on the associated server. However, this action is not required.

Policy

Every high availability group has an associated policy. The policy is used to determine which members of a high availability group are active at a given point in time. The policies available for high availability groups to use are stored as part of the core group configuration. See “High availability group policies” for more information.

High availability group policies

Every high availability group has a policy associated with it. This policy is used to determine which members of a high availability group are active at a given time.

The policies that the high availability groups use are stored as part of the core group configuration. The same policy can be used by several different high availability groups, but all of the high availability groups to which it applies must be part of the same core group. Before modifying or deleting an IBM provided policy, see “High availability group policy modification guidelines” on page 459

Policy selection

Policies are statically configured, and the high availability groups that are governed by them are created dynamically. Therefore, a mechanism is required to associate a running high availability group to a configured policy. This association is accomplished by comparing the following two pieces of information.

- The high availability group name
- The policy match criteria

“High availability group policy selection process” provides more detail about how a policy is selected.

Policy settings

Some policy settings apply for all policy types while others only apply for specific policy types. Some of the policy settings also influence the overall behavior of a policy. “Implications of high availability group policy settings” on page 456 describes the various settings, the applicable policy types, and how they influence policy behavior.

Policy enforcement

Whenever one of the following conditions occurs, the high availability manager runs the policy that is associated with a high availability group and takes any appropriate action:

- A member joins or leaves that high availability group. A member leaves the group if the member fails.
- The state of a member of that high availability group changes. For example, if the state changes from idle to active, or from idle to disabled, the policy rules are reapplied.

Policy changes

The high availability manager dynamically detects policy configuration changes. Therefore, policy setting changes go into affect as soon as you save and propagate these changes. Server restarts are not required.

High availability group policy selection process

Every high availability group has a unique group name that consists of a set of name-value pairs. Every policy definition contains an attribute called *match criteria* that is also a set of name-value pairs. To determine the policy for a high availability group, the group name is compared to the match criteria of all the associated core group polices. The policy with the strongest match to the group name is assigned to the high availability group:

When selecting a policy for a high availability group, the high availability manager:

1. Finds the set of policies that are eligible to govern a high availability group. For a policy to be eligible, all name-value pairs in the match criteria of an eligible policy must be contained in the name of the high availability group.
2. Selects the policy that has the most name-value pair matches from the list of eligible policies, and uses that policy to govern the high availability group.

Any component can create a high availability group for that component to use. However, the component code must specify the name-value pairs that are used for the high availability group name. The WebSphere Application Server administrator can control the name-value pairs that make up a policy match criteria, and thereby control which policy governs a particular high availability group.

WebSphere Application Server includes a couple of predefined policies. The following examples demonstrate the matching mechanism that is used for these policies.

Clustered TM Policy

The transaction manager component uses the policy Clustered TM Policy when the component is configured for high availability. The following description illustrates why, under these conditions, this policy is selected for the transaction manager high availability group:

- A cluster member process, such as ServerA, is started.
- The transaction manager component code joins a high availability manager to the high availability group named:

```
GN_PS=testCell\testNode\ServerA,IBM_hc=MyCluster,type=WAS_TRANSACTIONS
```

- ServerA is defined as a member of the DefaultCoreGroup core group for which the following policies are defined:
  - Clustered TM Policy, which has the match criteria type=WAS_TRANSACTIONS.
  - Default SIBus Policy, which has the match criteria type=WSAF_SIB.
- The high availability manager compares the group name to the match criteria for the two available policies. The high availability manager eliminates the Default SIBus Policy because the match criteria is not a proper subset of the high availability group name. The high availability manager determines that Clustered TM Policy is the closest match because:
  1. The match criteria for that policy includes the name-value pair type=WAS_TRANSACTIONS, which is also specified in the high availability group name. Therefore, the match criteria is a proper subset of the high availability group name.
  2. The match criteria for that policy more matches (one) than the match criteria for Default SIBus Policy, which is eliminated because it does not have any matches.

Administrator TM Policy

This example builds on the previous example and demonstrates how an administrator can define a new policy to govern the transaction manager high availability group. In this example the same high availability group name and default policies that are described in the previous example are used. However, in this example, the administrator creates a new policy in the DefaultCoreGroup configuration called the Administrator TM Policy. For the high availability manager to select this new policy, the policy must be eligible and contain more matches than any other policy.

The following description illustrates why, under these conditions, the policy Administrator TM Policy is selected for the transaction manager high availability group:

- The cluster member process ServerA is started.
- The transaction manager component code joins a high availability manager to the high availability group named:

```
GN_PS=testCell\testNode\ServerA,IBM_hc=MyCluster,type=WAS_TRANSACTIONS
```

- ServerA is defined as a member of the DefaultCoreGroup core group, for which the following policies are defined:
  - Clustered TM Policy, which has the match criteria type=WAS_TRANSACTIONS.
  - Default SIBus Policy, which has the match criteria type=WSAF_SIB.
  - Administrator TM Policy, which has the match criteria IBM_hc=MyCluster,type=WAS_TRANSACTIONS.
- The high availability manager compares the group name to the match criteria for the available policies. The high availability manager eliminates the Default SIBus Policy because the match criteria is not a proper subset of the high availability group name. It determines that Clustered TM Policy and Administrator TM Policy are both eligible policies, because their match criteria are proper subsets of the high availability group name:
  - Clustered TM Policy contains the name-value pair type=WAS_TRANSACTIONS, which is also specified in the high availability group name.
  - Administrator TM Policy contains the name-value pairs IBM_hc=MyCluster and type=WAS_TRANSACTIONS, which are both specified in the high availability group name.

  Because Administrator TM Policy has two matching pairs, IBM_hc=MyCluster and type=WAS_TRANSACTIONS, and Clustered TM Policy has only one matching pair, type=WAS_TRANSACTIONS, the high availability manager associates Administrator TM Policy with the transaction manager high availability group.

Ambiguous Matches

Do not configure identical match criteria for multiple policies in the same core group. Configuring identical match criteria causes an ambiguous match to the associated high availability group. Because a high availability group can only be associated with one policy, if the previously described matching mechanism does not result in a single policy match, the high availability manager puts the high availability group in error state, and does not make any of the group members active. Depending on the nature of the problem, the high availability manager might write one of the following error messages to the SystemOut.log file:

```
HMGR0301W: No policy was located for the group named {0}
HMGR0302W: Multiple policies match the group named {0}, Matching Policies are {1}
```

You can use the administrative console to view the policies associated with a high availability group and the current state of members of that group.

Implications of high availability group policy settings

All of the settings that are specified for a policy affect how the high availability manager governs a high availability group associated with that policy. Some of the policy settings are policy type specific, while others apply to all policy types. It is important to understand the implications for all of the associated high availability groups before you change the settings of an existing policy.

Implications of the Policy type setting

The policy type determines which members of a high availability group are automatically made active when the servers containing these members start. You can not directly change the policy type of an existing high availability group policy. If you need to change the policy type, you must create a new policy with a different policy type and give it a match criteria that makes the high availability manager select a new policy instead of the original one to associate with the high availability group.

Before creating a new policy with a different policy type, you must determine which components are using the high availability groups that are governed by the original policy and make sure that those components support the new policy type. For example, the service integration bus (SIB) component might require a One of N policy for its high availability group, because it only wants one group member active at a given ime. If you change the policy that is associated with the service integration bus high availability group to be an All Active policy, the service integration bus high availability support might not function properly and data corruption can occur.

You can select one of the following policy types when you create a new policy:

All active policy
When this policy is selected, all of the members of the high availability group are made active.

M of N policy
When this policy is selected for a high availability group with N members, M of them are made active. The number that M represents is configurable in the policy settings. You can use the Preferred servers setting to designate which members of the group to activate first.

No operation policy
When this policy is selected, none of the high availability group members are made active. You can use the administrative console to manually activate specific group members.

One of N policy
When this policy is selected for a high availability group with N members, only one member of the group is made active. You can use the Preferred servers setting to designate which member of the group are made active.

Static policy
When this policy is selected, only the members specified in the Static group servers setting are made active.

Implications of the Preferred servers setting

With the One of N and M of N policy types, you can set up a list of preferred servers as part of the policy settings. The preferred server list enables an administrator to indicate a preference as to which high availability group member is made active. If no preferred server list is specified, any of the available high availability group members can be selected as the member to activate. If a preferred server list is specified, then the member to activate is selected from this list, in order of preference. The most preferred server is the first one on the list. The following example demonstrates how a policy uses of the preferred server list.

Example:

A high availability group has three members that are located on application servers named ServerA, ServerB, and ServerC. This group is governed by a One of N policy, under which only one of the three members can be active at a given time. When all three members are running and available at the time that the policy is enforced:

- If no preferred servers are specified, the high availability manager randomly selects one of the three members and makes it active.
- If ServerB is the only server on the preferred servers list, the high availability manager makes the member located on this server active before either of the other two members, provided the member located on this server is available when the policy is enforced.
- If all three of the application servers are listed on the preferred servers list in the following order, and if all other things are equal, the high availability manager makes the member that is located on ServerC active:

  ServerC

  ServerA

  ServerB

The two other policy settings that directly affect how the preferred server list is used are the Failback and Preferred servers only settings.

Implications of the Failback setting

The Failback setting is used to specify what happens to the high availability group member on the most preferred server when it is restarted following a failure. The affect of the Failback setting on a member is best demonstrated with two examples.

During startup example:

A high availability group has three members that are located on application servers named ServerA, ServerB, and ServerC. This group is governed by a One of N policy, under which only one of the three members can be active at a given time. The server named ServerB is the only server on the preferred server list. In this example, none of the servers are started.

When ServerA starts, the One of N policy dictates that the high availability manager makes a member active. Because this application server is the only server running, the member on ServerA is activated. When we start ServerB, which is the only server on the preferred server list, one of two things happens:

- If Failback is enabled when ServerB starts, the high availability manager deactivates the currently active member and activates the member on ServerB, because ServerB is on the preferred server list.
- If Failback is disabled when ServerB starts, the currently active member remains the active member.

Following a failure example:

A high availability group has three members that are located on application servers named ServerA, ServerB, and ServerC. This group is governed by a One of N policy, under which only one of the three members can be active at a given time. ServerB is the only server on the preferred server list and is the only member that is currently active. If ServerB fails, the high availability manager activates one of the remaining members to replace that member. The Failback setting determines what happens after ServerB is repaired and restarted.

- If Failback is enabled when ServerB restarts, the currently active member is deactivated, and the member on ServerB is activated, because ServerB is still the most preferred server
- If Failback disabled when ServerB restarts, the currently active member remains the active member.

Implications of the Preferred servers only setting

The Preferred Servers Only setting is used to instruct the policy to activate members on preferred servers only. With this setting enabled, only members running on the servers that are specified in the preferred servers list are activated. If no preferred servers are specified, or no preferred servers are currently available, then no members are activated.

During startup example:

A high availability group has three members that are located on application servers named ServerA, ServerB, and ServerC. This group is governed by a One of N policy, under which only one of the three members can be active at a given time. ServerB the only server on the preferred server list. In this example, none of the servers are started.

When ServerA starts, the One of N policy dictates that the high availability manager activates a member. Because ServerA is the only server running, the member on ServerA is activated. Because it is the only server on the preferred server list, when ServerB starts, one of two things happens:

- If Preferred servers only is enabled when ServerA or ServerC starts, no member is activated because the high availability manager can only activate a member that is located on a server that is on the preferred servers list. When ServerB starts, the high availability manager activates the member on ServerB because ServerB is on the preferred servers list.
- If Preferred servers only is disabled when ServerA starts, the member on ServerA is activated because any member of the group can be the active member. When ServerB or ServerC starts, no activation occurs because the member on ServerA is already active.

Following a failure example:

A high availability group has three members that are located on application servers named ServerA, ServerB, and ServerC. This group is governed by a One of N policy, under which only one of the three members can be active at a given time. ServerB the only server on the preferred server list. The member located on ServerB is the active member. If ServerB fails, one of two things happens:

- If Preferred servers only is enabled when ServerB fails, the high availability manager can only activate another members that is located on a server that is included on the preferred servers list. Because ServerB is the only server on the preferred servers list, no other member is activated.
- If Preferred servers only is disabled when ServerB fails, the high availability manager activates one of the remaining members to replace the member on ServerB.

Implications of the Static group servers setting

You can specify a list of static group servers as part of the configuration settings for a static policy type. When a high availability group is governed by a static policy type, the static group server list defines which group members are activated if it is possible to do so.

Implications of the Is alive timer setting

The Is alive timer setting controls how frequently the high availability manager checks the health of the active group members that are governed by a given policy. The high availability manager can detect two fundamentally different kinds of failures:

- It can detect when an entire process stops functioning or terminates. This type of failure detection does not depend on the value that is specified for the Is alive timer setting.
- It can detect when a program fails. This type of failure detection does depend on the value that is specified for the Is alive timer setting. The value that is specified for the Is alive time setting determines the amount of time that might pass before a processing problem that does not cause the entire process to stop functioning or terminate is detected.

The administrator has the ability to specify the Is alive timer at the policy level, where it applies to all the members that are governed by this policy, at the process level where it applies to all members running in a particular process. The administrator can also disable this type of failure detection at either of these levels.

Implications of the Quorum setting

Quorum is a mechanism that you can use to protect resources that, in the event of a failure, are shared across members of a high availability group. When enabled, the policy does not activate any group members until quorum is achieved. A high availability group does not achieve quorum until a majority of the members are running. For example, if there are n members in a group, (n/2) + 1 servers must be online to achieve quorum.

Quorum is an advanced function that is designed to work with clusters, specialized component code, and a hardware control facility. Currently, none of the high availability groups supporting WebSphere Application Server components use the quorum mechanism. Therefore, do not enabled the Quorum setting.

High availability group policy modification guidelines

Use the following guidelines to help determine when to create a new high availability group policy, and when to modify or delete an existing policy.

Do not delete the default IBM provided policies

If you want to override one of the default policies that IBM provides, it is recommended that you do not delete the current policy. Instead, create a new policy with more specific match criteria. The policy with the greatest number of matches is the one that is used. Not deleting the IBM provided policy enables you to revert back to that policy if a problem occurs with your new policy.

Do not try to change the type of an existing policy

After a policy for a high availability group is created, you can change some of the policy attributes such as preferred servers, or failback, but you cannot change the policy type. If you need to change the policy type, you must create a new policy and then use match criteria to associate it with the appropriate high availability group. See “High availability group policy selection process” on page 454 for a description of how the high availability manager selects a policy for a high availability group.

Make sure that you know which policies a component using that high availability group supports before creating a new policy to associate with that group

A component does not necessarily support all policy types and options. Therefore, before changing the policy that is associated with a given high availability group, make sure you fully understand if the application server code using that high availability group supports the change. For example, if you want to change the type of policy that is associated with the high availability group used by the transaction manager component, make sure the transaction manager code supports the new policy type before making the change.

Do not use the same match criteria for multiple policies in the same core group

If you have multiple policies configured with identical match criteria, the policy match to the associated high availability group is ambiguous. If you are creating a new policy to replace an older policy that you created, you might need to delete the older policy to specify the appropriate match criteria. Another alternative is to specify additional match criteria in each policy so no ambiguity exists as to which policy is controlling the high availability group.

Changing the number of core group coordinators

For a large cell, multiple coordinators are recommended.

Before you change the number of core group coordinators for a cell:

- Make sure that you are familiar with the content of the “Core group coordinator” on page 442 topic.
- Determine the number of coordinators for the core group. A general guideline is approximately one coordinator per 20 core group members.

Change the number of core group coordinators only if:

- IBM support instructs you to do so.
- You are directed to do so as part of the installation process for another WebSphere product.
- The coordinator process is consuming abnormally high amounts of memory or CPU resources.
- You are scaling up the number of servers in a cell.

To change the number of core group coordinators for a core group:

1. In the administrative console, click **Servers > Core groups > Core group settings** and select an existing core group.
2. In the **Number of coordinators** field, specify the number of coordinators that you want for this core group.
3. Click **OK**, and then click **Save**.
4. Click **Synchronize changes with nodes** and then click **Save** again to save your changes.
5. If you need to change the list of preferred coordinator servers, click **Preferred coordinator servers** and then add or delete the appropriate application servers from the list of preferred coordinator servers.
6. Click **OK** and then click**Review**.
7. Select **Synchronize changes with nodes**, and then click **Save**.

Your changes take effect immediately after synchronization completes. The members of the core group pick up these changes dynamically.

Core group settings

Use this page to create a core group or to edit an existing core group. A core group is a component of the high availability manager function. It can contain standalone servers, cluster members, node agents and the deployment manager. A core group must contain at least one node agent or the deployment manager.

Before you create a core group you must understand the relationship of core groups in a high availability environment and know how you intend to use each core group.

To view this administrative console page, click **Servers > Core groups > Core group settings > New** or *Select an existing core group for editing*.

On the **Configuration** tab, you can edit fields. On the **Runtime** tab, you can look at read-only information.

After you specify your core group settings, click **Apply** before defining additional properties or setting up a core group bridge.

Extended information about the core group fields:

Name

Specifies the name of the core group. This field can only be edited when you create new core groups.

If you are defining a new core group, specify a name that is unique among the existing core groups. It is helpful to other WebSphere Application Server administrators if the name helps define the use of this core group and if it is consistent with the names of the other core groups in the cell.

This field can contain alpha and numeric characters. The following characters cannot be used in this field:

```
# \ / , : ; " * ? < > | = + & % '
```

Also, the name cannot begin with a period (.) or a blank space. A blank space does not generate an error. However, leading and trailing blank spaces are automatically deleted.

For example, `DefaultCoreGroup` is the name of the core group that contains the deployment manager server process.

Description

Specifies a description of the core group. In environments where there are multiple system administrators this field can help these administrators understand the overall organization of the core groups. The supported length of this field is quite large. However, long descriptions take time to load and can cause a delay when displaying the page.

Example: ″Default Core Group. The default core group cannot be deleted.″ is the description of the DefaultCoreGroup.

Number of coordinators

Specifies the number of coordinators for this core group. The coordinator is the aggregation point for high availability manager information. The coordinator determines group membership and communicates state and status to the other members of the core group.

The default value is one coordinator, although multiple coordinators are advisable for large core groups. All of the group data must fit in the memory of the allocated coordinators. One coordinator can run out of memory in a system with a large core group, which can cause the system to work improperly.

Transport type

Specifies the transport mechanism to use for communication between members of a core group. This is a required field.

Channel framework
Channel framework is the default transport type. It uses the channel framework service to incorporate port reusability and shared port technology into the communication system.

Unicast
Unicast is a targeted network model that focuses on a direct recipient for communication. This type of communication is most suitable when the intended message is sent to a specific set of recipients.

Multicast
Multicast consists of a broadcast network model. This model broadcasts communication across the defined network, depending upon the values that are provided for the multicast settings. Multicast

settings are suitable when there are many recipients for the intended message; otherwise broadcast communication tends to overload the network with traffic, and can impact performance goals.

Important: If a core group needs to use the core group bridge service, you must select Channel framework as the transport mechanism for that core group. If you select Unicast or Multicast, you might receive error message CHFW0029E, which indicates that the transport chain could not be initialized because the address was already in use.

Channel chain name

Specifies the name of the channel chain if you select channel framework for the transport type.

Multicast settings

Specifies the following settings for a multicast transport type. These settings are only valid if you select multicast for the transport type:

- **Multicast port**

  The port setting tells the coordinator where to scan for transmissions. When setting this value, verify that you are specifying a port that is not used by another network communication device. Setting a port value that has conflicts causes problems with your high availability manager infrastructure.
- **Multicast group IP start**

  Specify the starting Internet Protocol (IP) address of the intended communication area.
- **Multicast group IP end**

  Specify the ending IP address of the intended communication area. Plan the network to accommodate scalability.

Additional properties

Core group servers

Specifies the server processes that belong to the core group. Server processes include the deployment manager, node agents, application servers, and cluster members. You can use the panel that displays to move server processes to a different core group.

Custom properties

Specifies the custom properties that are used for configuration purposes.

Policies

Use to define the policies that determine which members of a high availability group are made active.

Preferred coordinator servers

Specifies which core group servers are preferred coordinator servers.

Related topics

Core group bridge settings

Click on to specify core group bridge communication settings between core groups.

Group name properties

Specifies one or more name=value pairs as the match criterion for a high availability group. If you specify more than one name=value pair, use a comma to separate the pairs. You can specify an asterisk (*) to obtain the selected information for all of the high availability groups within this core group.

When a WebSphere Application Server component creates a new high availability group, it establishes a map of that group's properties as the group name. This map is used to uniquely identify that high availability group.

After you specify a match criterion or an asterisk:

- Select **Calculate** to determine how many high availability groups have names that match this match criterion.
- Select **Show groups** to view a list of the currently running high availability groups that match this match criterion. For each group, this list indicates:
  - Its high availability group name
  - Whether or not quorum has been enabled
  - The policy that is associated with the high availability group. If more than one policy is listed for a high availability group, you must change the match criterion for one or more of your policies so that only one policy is associated with this high availability group.
  - Its status (either the OK icon or the Error icon). If only one policy is listed in the Policy column, the OK icon is displayed in the Status column. If more than one policy is listed Policy column, the Error icon is displayed in the Status column.
- Select **Show severs** to view a list of servers which are hosting active members of the high availability groups that match the specified Group name properties. For each server, this list indicates:
  - The names of the servers which are hosting the active high availability group members.
  - The name of the node on which these servers resides.
  - The version of the WebSphere Application Server product on which these servers are running.
  - The number of high availability group members that are currently active on these servers.

Example: Suppose the following high availability groups are defined for a core group:

- Component A uses the following properties for its group name: [name=compA, policy=oneofN, owner=smith]
- Component B uses the following properties for its group name: [name=compB, policy=MofN, owner=smith]
- Component C uses the following properties for its group name: [name=compC, policy=oneofN, owner=smith]

If you specify `policy=oneofN` in the **Group name properties** field and then select **Show groups**, the groups for components A and C are listed.

If you specify `owner=smith` in the **Group name properties** field and then select **Show groups**, the groups for components A, B and C are listed.

If you specify all of component C’s name properties in the **Group name properties** field:

```
name=compC,policy=oneofN,owner=smith
```

Then select **Show groups**, only the group for component C is listed. Note that the properties are separated by commas. There are no blank spaces.

Core group custom properties

Use these custom properties for advanced configurations for core groups.

To configure a custom property, complete the following steps:

1. In the administrative console, click **Servers > Core groups**, and then select one of the listed core groups.
2. Under Additional Properties, click **Custom properties**
3. If the custom property is in the list of defined custom properties, click on that property and then enter an appropriate value in the Value field.

   If the custom property is not in the list of defined custom properties, click **New** and then enter the name of the custom property in the Name field and an appropriate value in the Value field.

IBM_CS_LS_DATASTACK_MEG

Use this custom property to eliminate a condition that is reported by a message that is displayed repeatedly in your `SystemOut.log` file.

You might see a message, similar to the following message, repeated multiple times in the SystemOut.log file:

```
[9/24/04 13:28:19:497 CDT] 00000013 VSync         W
DCSV2005W: DCS Stack DefaultAccessPointGroup.P at Member 172.16.1.86:9353:
The amount of memory available for synchronization is low. The configured memory
size is 418816 bytes. Currently used memory size is 420307 bytes.
```

If the member IP address is in the format of a dotted decimal IP address and port, you can eliminate these messages by increasing the amount of memory that is allocated to the core stack that is used for core group communication. Increase the value of this property until you no longer see the message in your SystemOut.log file. Because the memory is dynamically allocated, setting a larger stack size than you need does not cause memory problems.

Note: You can also set this custom property:

- On the core group bridge interface that contains the particular core group member that is in the messages.
- On the access point group or the core group access point for the particular core group member that is in the messages.

See “Core group bridge custom properties” on page 517 for more information about how to set the property at those levels.

Units	megabytes
Default	5

IBM_CS_IP_REFRESH_MINUTES

Use this custom property to adjust how frequently the core group IP cache is cleared.

The caching of name-to-IP address information at the core group level eliminates some of the system overhead required to assign IP addresses to core group members.

Units	seconds
Default	60
Acceptable values	Any positive integer. 1 is the minimum value that can be specified

IBM_CS_UNICAST_DISCOVERY_INTERVAL_SECS

Use this custom property to change how frequently the high availability manager Discovery Protocol checks for new core group members. A new core group member is not able to communicate with other core group members until the Discovery Protocol establishes communication between the new member and the existing members.

See “Core group Discovery Protocol” on page 449 for more information about the Discovery Protocol.

Units	seconds
Default	30 seconds.

IBM_CS_FD_PERIOD_SECS

Use this custom property to change how frequently the Failure Detection Protocol checks the core group network connections that the discovery protocol establishes. The Failure Detection Protocol notifies the Discovery Protocol if a connection failure occurs.

See “Core group Failure Detection Protocol” on page 449 for more information about the Failure Detection Protocol.

Units	seconds
Default	6

IBM_CS_SOCKET_BUFFER_SIZE

Use this custom property to change the size of the socket buffer that the core group transport obtains.

“Configuring core group socket buffers” on page 474 includes a table that shows the relationship between the values that can be specified for this property and the underlying memory allocation size per socket buffer type.

Units	One of the following: • No over rides • small • medium • large
Default	2 megabytes for all buffer types.

Configuring core group preferred coordinators

The high availability manager uses an ordered list of preferred core group servers when it selects servers to host the coordinators. If the default list is inappropriate, you can change the list such that other servers are selected as coordinators.

Use the following guidelines to determine the appropriate ordered list of preferred coordinators:

- Make sure that you are familiar with the content of the “Core group coordinator” on page 442 topic.
- Select a set of servers that infrequently start and stop, are stable, and that are located on capable systems with large amounts of system memory and a fast processor.

You might want to change the ordered list of preferred core group servers if:

- The system where the default coordinator is located has insufficient resources or capacity to serve as a coordinator.
- A server that normally gets selected as the coordinator starts and stops frequently.

To change the ordered list of preferred core group servers:

1. In the administrative console, click **Servers > Core groups > Core group settings** and select an existing core group.
2. Click **Preferred coordinator servers** and then add or delete the appropriate application servers from the list of preferred coordinator servers. Use **Add** and **Remove** to move servers into and out of the list of core group servers on which the coordinator service can run. Use **Move up** and **Move down** to adjust the order of the servers within this list. Make sure that the most preferred server is at the beginning of the list and the least preferred server is at the end.
3. Click **OK** and then click**Review.**
4. Select **Synchronize changes with nodes**, and then click **Save**.

Your changes take effect immediately after synchronization completes. The members of the core group pick up these changes dynamically.

Preferred coordinator servers settings

Use this page to define an ordered list of preferred core group servers. This list indicates where the high availability manager coordinators will run.

To view this administrative console page, click **Servers** > **Core groups** > **Core group settings** > **New** > or *select an existing core group* > **Preferred coordinator servers**.

Use **Add** and **Remove** to move servers into and out of the list of core group servers on which coordinator service will run. Use **Move up** and **Move down** to adjust the order within this list. Make sure that the most preferred server is at the top of the list and the least preferred server is at the bottom.

Click **OK** to make your changes effective. Click **Save** to save and synchronize your changes with all managed nodes.

Configuring a core group transport

When you configure a core group, you can specify the type of network transport that you want the high availability manager to use for network communications.

You should do the following before you start to configure a core group transport:

- Make sure that you are familiar with the content of the “Core group transports” on page 447 topic.
- Confirm that all node agents in the core group are running.
- Determine which transport type is appropriate for the core group.

You need to change the transport setting for a core group if the current transport type does not meet your needs. For example, you might need to increase security, you might want to maximize replication throughput, or a core group might need to use the core group bridge service.

Important: If a core group needs to use the core group bridge service, you must select Channel framework as its transport mechanism. If you select Unicast or Multicast, you might receive error message CHFW0029E, which indicates that the transport chain could not be initialized because the address was already in use.

To change the core group transport for a core group:

1. In the administrative console, click **Servers** > **Core groups** > **Core group settings** and select an existing core group or click **New** if you are creating a new core group.
2. Under Transport type, select the type of transport that you want to use for this core group. You can select one of the following types of transports:
   - Channel framework, which is the default transport type. Channel framework is a flexible point-to-point transport. If you choose this transport type, you must also select one of the transport chains that is listed in the Channel chain name field.
     – If you select DCS, the high availability manager performs related network communication over non-secure sockets.
     – If you select DCS-Secure, the high availability manager performs related network communication over Secure Sockets Layer (SSL) encrypted sockets.
   - Unicast, which is a point-to-point transport over standard non-secure operating system-level TCP/IP network connection facilities.
   - Multicast, which is an IP multicast transport. This type of transport broadcasts all information to all of the other processes in the core group. If you choose this transport type, you must also specify:
     – In the Multicast port field, the port number for this multicast transport. The default port is 23445.

- In the Multicast group IP start field, the first IP address in the range of IP addresses to use for this multicast transport . The default IP address is 239.0.0.0
- In the Multicast group IP end field, the last IP address in the range of IP addresses to use for this multicast transport . The default IP address is 239.255.255.255.

3. Click **OK**, and then click **Save**.
4. Select **Synchronize changes with nodes**, and then click **Save** again.
5. Restart the servers containing core group members.

After the servers complete their restart, they all use the new transport.

Interoperating with Version 6.0.1.2 processes

The high availability manager supports multihomed hosts, which means that WebSphere Application Server processes can communicate with each other even if they are running on different versions of the product. If you are running Version 6.0.1.2 processes that need to communicate with each other, there are high availability manager-related issues that you need to consider.

Know the version levels of the processes that need to communicate with each other. If no V6.0.1.2 processes exist, see “Interoperating with Version 6.0.2 and later processes” on page 468.

In Version 6.0.1 without Fix Pack 2 installed, an acceptable value for the DCS_UNICAST_ADDRESS host name field is either a host name, for example, myhost.mydomain, or a textual IP address, for example, 192.168.0.2. If a host name is specified, Domain Name Services (DNS) is used to resolve the host name to an IP address. If the host supports multiple IP addresses, where the host name has multiple mappings in DNS, a host name is ambiguous and a textual IP address is required.

After the installation of Version 6.0.1 Fix Pack 2, the asterisk is recognized as a valid value for the DCS_UNICAST_ADDRESS host name field. When this field is set to an asterisk, multihome support allows the high availability manager to open and receive connections on all IP addresses available for the host.

With the introduction of Version 6.0.1 Fix Pack 2, processes can be classified into three separate categories:

Type 1
Processes that can operate only in single-IP mode. This category includes processes on Version 6.0 nodes or Version 6.0.1 nodes that do not have Version 6.0.1 Fix Pack 2 installed.

Type 2
Processes on Version 6.0.1 nodes that have Version 6.0.1 Fix Pack 2 installed, but do not have the DCS_UNICAST_ADDRESS host name field for the process set to an asterisk.

Type 3
Processes on Version 6.0.1 nodes that have Version 6.0.1 Fix Pack 2 installed and have the DCS_UNICAST_ADDRESS host name field for the process set to an asterisk.

The following interoperability rules apply to these process types:

- Type 1 and type 2 processes can interoperate.
- Type 2 and type 3 processes can also interoperate.
- Type 1 and type 3 processes cannot interoperate.

  Therefore, careful planning is necessary before converting processes to type 3 processes.

To enable interoperating with Version 6.0.1.2 processes:

1. Add multihome capability by installing Version 6.0.1 Fix Pack 2. This step changes the existing type 1 processes to type 2 processes.

   a. Apply Version 6.0.1 Fix Pack 2 to all existing nodes. Nodes on which Version 6.0.1 Fix Pack 2 is installed continue to interoperate with nodes on which this fix pack is not installed, as long as the configuration does not change.
2. Stop all application servers.
3. Ensure that the deployment manager and all of the node agents are running.
4. Change the DCS_UNICAST_ADDRESS Host name field of each process to an asterisk. This step changes the existing type 2 processes to type 3 processes.
   a. In the administrative console, go to the configuration data for one of the processes on the new nodes:
      - For an application server, click **Servers > Application Servers >** *server_name*, and then, under Additional Properties, click **> Ports >** *port_name* .
      - For a node agent, click **System administration > Node agents >** *node_name*, and then, under Additional Properties, click **> Ports >** *port_name* .
      - For a deployment agent, click **System administration > Deployment manager**, and then, under Additional Properties, click **> Ports >** *port_name* .
   b. Click **View associated transports** for the port that is associated with the DCS transport channel that you want to review.
   c. Click **DCS_UNICAST_ADDRESS**, and enter the name of the new host in the Host field.
   d. Click **OK**, and then click **Save**.
   e. Repeat the previous steps until the new host name is added for all of the processes on the new nodes.
   f. Select **Synchronize changes with nodes**, and then click **Save** again.
5. Stop all of the servers that contain the processes with configuration changes.
6. Restart these servers.

All of the processes can communicate with each other.

After multihome support is enabled, all of the new nodes that are added to the installation must have multihome support enabled. The base Version 6.0.1 code and Version 6.0.1 Fix Pack 2 must be installed before profiles are created and the node is added. If this procedure is not observed, manual configuration is required to enable the processes to connect.

Your configuration cannot contain a mix of type 1 and type 3 processes. To ensure a valid configuration:

1. Check for type 1 processes. Type 1 processes log the following message, which indicates that the host name for another process in the core group is an asterisk:

   ```
   HMGR0024W: An error was encountered while looking up the IP address for
   the host name of a core group member. The host name is * and the server
   name is myCell01\myCellManager01\dmgr. The member will be excluded from
   the core group.
   ```
2. Check for type 3 processes. Type 3 processes do not display in a view with type 1 processes, but can be detected by examining the HMGR0218 messages the various processes log. If the processes connect, the same message is logged across all processes. Specifically, processes that connect have the same view identifier and the same number of processes in the view.

   ```
   HMGR0218I: A new core group view has been installed. The core group is
   defaultCoreGroup. The view identifier is (8:0.spoletoCell01\
   spoletoCellManager01\dmgr). The number of members in the new view is 2.
   ```

Interoperating with Version 6.0.2 and later processes

The high availability manager supports multihomed hosts, which means that WebSphere Application Server processes can communicate with each other even if they are running on different versions of the product. If you are running Version 6.0.2 and later processes that need to communicate with each other, you need to consider there are high availability manager-related issues.

Know the version levels of the processes that need to communicate with each other. If Version 6.0.1.2 processes exist, see "Interoperating with Version 6.0.1.2 processes" on page 467 for additional considerations.

When processes are created on Version 6.0.2 or later nodes, the host name field in the DCS_UNICAST_ADDRESS endpoint is set to an asterisk. When you use an asterisk in the host name field, multihome support allows the high availability manager to open and accept connections using any IP address that is valid for that machine.

Special considerations are required when Version 6.0.2 nodes interoperate with Version 6.0 and Version 6.0.1 nodes without Fix Pack 2 (6.0.1.2), because these earlier versions do not contain high availability manager multihome support. Version 6.0 and Version 6.0.1 processes do not recognize an asterisk as a valid value in the DCS_UNICAST_ADDRESS host name field. Therefore, these processes cannot connect to Version 6.0.2 processes that are configured with an asterisk in the DCS_UNICAST_ADDRESS host name field. When Version 6.0.2 processes must interoperate with Version 6.0 or Version 6.0.1 processes, the DCS_UNICAST_ADDRESS for all Version 6.0.2 processes must be configured to use a value other than an asterisk. This configuration is done by setting the host name field in the DCS_UNICAST_ADDRESS endpoint to a host name or to a textual IP address.

- If the host is configured to use a single IP address, a string host name, for example, myhost.mydomain, is sufficient.
- If the host is configured to use multiple IP addresses, then a textual IP address, for example, 192.168.0.2, is required.

As you add Version 6.0.2 nodes and create servers into a mixed-release cell, you must set the host name field of the DCS_UNICAST_ADDRESS for all processes on the new nodes, including the node agent, to one of the two values previously specified. You must then restart the Version 6.0.2 processes to pick up the new value.

To change the host name field for a process:

1. In the administrative console, go to the configuration data for one of the processes on the new nodes:
   - For an application server, click **Servers > Application Servers >** *server_name*, and then, under Additional Properties, click **> Ports >** *port_name* .
   - For a node agent, click **System administration > Node agents >** *node_name*, and then, under Additional Properties, click **> Ports >** *port_name* .
   - For a deployment agent, click **System administration > Deployment manager**, and then, under Additional Properties, click **> Ports >** *port_name* .
2. Click **View associated transports** for the port that is associated with the DCS transport channel you want to review.
3. Click **DCS_UNICAST_ADDRESS**, and enter the name of the new host in the Host field.
4. Click **OK**, and then click **Save**.
5. Repeat the previous steps until the new host name is added for all of the processes on the new nodes.
6. Select **Synchronize changes with nodes**, and then click **Save** again.
7. Stop all of the servers that contain the processes with configuration changes.
8. Restart these servers.

All of the processes can communicate with each other.

If Version 6.0.2 processes are not configured properly, the Version 6.0 and Version 6.0.1 processes might not start or might not be able to connect to the Version 6.0.2 processes.

The condition can be detected in one of the following ways:

- Version 6.0 and Version 6.0.1 processes log the following message:

```
HMGR0024W: An error was encountered while looking up the IP address for
the host name of a core group member. The host name is * and the server
name is myCell01\myCellManager01\dmgr. The member is excluded from the
core group.
```

This message indicates that the host name for another process in the core group is an asterisk.

- The Version 6.0, Version 6.0.1, and Version 6.0.2 processes form separate views if the Version 6.0 and Version 6.0.1 processes do not connect to Version 6.0.2 processes. To detect these views, examine the HMGR0218 messages that the various processes logged. If the processes connect, the same message is logged across all processes. Specifically, processes that are connected have the same view identifier and the same number of processes in the view.

```
HMGR0218I: A new core group view is installed. The core group is
defaultCoreGroup. The view identifier is (8:0.spoletoCell01\
spoletoCellManager01\dmgr). The number of members in the new view is 2.
```

Setting up IP addresses for high availability manager communications

There are situations where you must select a preferred IP address, or a range of IP addresses that you want the high availability manager to use for communication within a core group.

Determine the transport protocol the core group associated with the high availability manager uses for communications.

If the core group is configured to use a point to point protocol for its communications, for example, channel framework or unicast, you should only set a preferred IP address if you want to restrict the high availability manager communications to a specific IP address.

If the core group is configured to use a multicast transport for its communications, you must set a preferred IP address and specify a range of multicast group IP addresses.

1. Configure a preferred IP address
   a. Make a list of all WebSphere processes on this machine. Include both application server processes and administrative processes, such as node agents or the deployment manager, in this list. If a preferred IP is set for one process on a machine, it should be set for all processes on that machine
   b. Determine the textual form of the preferred IP address. For example, 9.5.87.124 or 10.1.2.2.
   c. Update the IP address specified for the for the DCS unicast address for all processes identified in the first substep with the textual form of the preferred IP address.
      1) In the administrative console, for an application server process click **Servers > Application servers**, for a node agent process click **System administration > Node agents**, or for a deployment manager process click **System administration > Deployment manager**. Then select the appropriate process.
      2) Under **Additional properties** , click on **Ports** to bring up the list of ports for the selected process and then click on the port named **DCS_UNICAST_ADDRESS**.
      3) Enter the preferred IP address in the Host field and then click **Apply** to save your changes.

      Following are the allowed values for the Host field:

      - * (an asterisk), which allows high availability manager communications on all NICs. This is the default value, but cannot be used if your processes must interact with V6 or V6.0.1 processes.
      - Text IP address, such as 10.1.1.2, which restricts high availability manager communications to the specified NIC. Text IP addresses can be used if the host is configured to use either a single IP address or multiple IP addresses.
      - A string host name of the form *myhost.mydomain* . A string host name cannot be used if the host is configured to use multiple IP addresses.
2. Specify a range of multicast group IP addresses for the high availability manager to use. If the core group transport is configured as multicast, you must associate a range of IP address with the NIC to

be used for high availability manager communications. You must use core group properties to specify this range of IP addresses. Do not specify them on a per process basis.

a. In the administrative console, click **Servers > Core groups > Core group settings**
b. Select the desired core group.
c. Specify the first IP address in the range associated with the NIC the high availability manager should use in the Multicast group IP start field
d. Specify the last IP address in the range associated with that NIC in the Multicast group IP end field.
e. Click **Apply** and then **Save** to save your changes.

The high availability manager uses the specified IP address or address range for communication within the core group.

Configuring the Discovery Protocol for a core group

The Discovery Protocol for a core group establishes network connectivity between a new core group member and the other members of that core group.

Understand the concepts that are described in “Core group Discovery Protocol” on page 449. Then determine how long you want the Discovery Protocol to wait before it recalculates the set of unconnected core group members and attempts to open connections to those members.

You might want to perform this task if you are trying to tune the discovery protocol behavior for a core group. The default value of 60 seconds, which is set during the WebSphere Application Server installation process, provides an acceptable process detection time for most situations.

To change wait interval for the Discovery Protocol:

1. In the administrative console, click **Servers > Core groups > Core group settings >** *core_group_name*.
2. Under Additional Properties, click **Custom Properties**.
3. Change the value that is specified for the IBM_CS_UNICAST_DISCOVERY_INTERVAL_SECS custom property.

   If the IBM_CS_UNICAST_DISCOVERY_INTERVAL_SECS property already exists, click on the property name and specify the new interval length, in seconds, in the Value field.

   If this property does not already exist, click **New** and create it:

   a. In the Name field, specify `IBM_CS_UNICAST_DISCOVERY_INTERVAL_SECS`.
   b. In the Value field specify the length of time, in seconds, you want the Discovery Protocol to wait before it recalculates the set of unconnected core group members and attempts to open connections to those members.
4. Click **OK** and then click**Review**.
5. Select **Synchronize changes with nodes**, and then click **Save**.
6. Restart all members of the core group.

After the servers restart, the core group members all run with the new Discovery Protocol setting.

Configuring the Failure Detection Protocol for a core group

The Failure Detection Protocol monitors the core group network connections that the Discovery Protocol establishes, and notifies the Discovery Protocol if a connection failure occurs.

- Understand the concepts that are described in “Core group Failure Detection Protocol” on page 449.
- Check your operating system settings that are relevant to TCP/IP socket closing events.

- Determine your failure detection goals and which settings must change to accomplish these goals.

You might want to perform this task if:
- You want to change the failover characteristics of your system.
- Your core groups are large and analysis indicates excessive CPU usage is spent monitoring heartbeats.

To change the settings for the Failure Detection Protocol:
1. In the administrative console, click **Servers > Core groups > Core group settings >** *core_group_name*.
2. Under Additional Properties, click **Custom Properties**.
3. Change the values specified for the IBM_CS_FD_PERIOD_SECS custom property. This property specifies the time interval, in seconds, between consecutive heartbeats. The default value for this property is 30 seconds.

   If the IBM_CS_FD_PERIOD_SECS property already exists, click on the property name, and in the Value field, specify the length of time, in seconds, that you want the Failure Detection Protocol to wait between consecutive heartbeats.

   If this property does not already exist, click **New** and create it:
   a. In the Name field, specify IBM_CS_FD_PERIOD_SECS.
   b. In the Value field specify the length of time, in seconds, that you want the Failure Detection Protocol to wait between consecutive heartbeats.
4. Change the values that are specified for the CS_FD_CONSECUTIVE_MISSED custom property. This property specifies the consecutive number of heartbeats that must be missed before the protocol assumes that the core group member has failed. The default value for this property is 6.

   If the CS_FD_CONSECUTIVE_MISSED properties already exists, click on the property name, and in the Value field, specify the number of heartbeats that must be missed before the Failure Detection Protocol assumes the core group member failed.

   If this property does not already exist, click **New** and create it:
   a. In the Name field, specify `CS_FD_CONSECUTIVE_MISSED`.
   b. In the Value field specify the number of heartbeats that must be missed before the Failure Detection Protocol assumes that the core group member failed.
5. Click **OK** and then click**Review**.
6. Select **Synchronize changes with nodes**, and then click **Save**.
7. Restart all of the members of the core group.

After the servers restart, the core group members all run with the new Failure Detection Protocol settings.

Configuring a core group for replication

WebSphere Application Server administrators can control the maximum amount of heap memory that the underlying core group transport can allocate. This memory is used for in-flight messages and network communication buffers. If you increase the maximum amount of heap memory that the transport can allocate, you must also increase the size of the transport buffer accordingly.

- Understand that other factors, such as the number of network interface cards on a machine, how the Network interface card is used, and network speed, can affect replication throughput performance.
- Determine the maximum amount of memory that you can let the core group transport allocate for buffering incoming messages. The default value is 10 megabytes. You can increase this value as required to allow for buffering of additional incoming messages. A setting of 100 is sufficient for most high throughput topologies.

You might want to perform this task if:
- You are trying to tune your systems replication performance.

- You are seeing large numbers of any of the following Distribution and Consistency Services (DCS) congestion messages in your SystemOut.log file:

  ```
  DCSV1051W, a high severity congestion event for outgoing messages
  DCSV1052W, a medium severity congestion event for outgoing messages
  DCSV1054W, a medium severity congestion event for incoming messages
  ```

  Important: Under extreme workloads, these messages might still occur on a properly tuned system.

To change amount of memory that is available for in-flight messages and network communication buffers:

1. Change the value of the IBM_CS_DATASTACK_MEG custom property. The value specified for the IBM_CS_DATASTACK_MEG custom property has a strong impact on the message throughput of a replication domain. The setting for this property controls the amount of memory that the data stack can use. The default value for this property is 50 megabytes. A replication domain that handles high throughput messaging, needs this property set to a higher value. The maximum value for this property is 256 megabytes. A setting of 100 megabytes is sufficient for most high throughput topologies.
   a. In the administrative console, click **Servers > Core groups > Core group settings >** *core_group_name*.
   b. Under Additional Properties, click **Custom Properties**.
   c. Change the value specified for the IBM_CS_DATASTACK_MEG custom property.

      If the IBM_CS_DATASTACK_MEG property already exists, click on the property name and, in the Value field, specify the maximum amount of memory that you want to let the core group transport allocate for buffering incoming messages.

      If this property does not already exist, click **New** and then:
      1) In the Name field, specify `IBM_CS_DATASTACK_MEG`.
      2) In the Value field, specify the maximum amount of memory you want to let the core group transport allocate for buffering incoming messages.
   d. Click **OK**, and then click **Save** to save your changes.
2. Change the size of the transport buffer. The maximum amount of heap memory that is specified for the IBM_CS_DATASTACK_MEG custom property should be less than or equal to the size of the transport buffer.
   a. In the administrative console, click **Servers > Application servers >** *server_name* **Core group service**.
   b. In the Transport buffer size field, specify, in megabytes the size of the transport buffer.
   c. Click **OK**, and then click **Save** to save your changes.
   d. Repeat this step for all of the core group members. Specify the same transport buffer size for all of the core group members.
3. Click **OK** and then click**Review**.
4. Select **Synchronize changes with nodes**, and then click **Save**.
5. Restart all members of the core group.

After the servers restart, they all run with the new replication settings.

Configuring core group IP caching

The caching of name-to-IP address information can be performed at many levels within the network communication software stack. These levels include within the operating system, the Java virtual machine, and within WebSphere Application Server components, such as core groups. Core groups use caching to reduce the overhead that is associated with IP address name lookup. You can adjust the interval at which a core group IP cache is cleared.

By default, a core group cache is cleared every 60 minutes. You can determine the correct time interval for your environment.

You want to change the length of time that IP addresses are retained in a core group cache.

To change how frequently a core group cache is cleared:

1. In the administrative console, click **Servers > Core groups > Core group settings >** *core_group_name*.
2. Under Additional Properties, click **Custom Properties**.
3. Change the value that is specified for the IBM_CS_IP_REFRESH_MINUTES custom property. A core group cache cannot be cleared more frequently then once a minute.

   If the IBM_CS_IP_REFRESH_MINUTES property already exists, click on the property name and in the Value field, specify the length of time, in minutes, you want to wait before a core group cache is cleared.

   If this property does not already exist, click **New** and create it:

   a. In the Name field, specify `IBM_CS_IP_REFRESH_MINUTES`.

   b. In the Value field specify the length of time, in minutes, that you want to wait before a core group cache is cleared.
4. Click **OK** and then click**Review**.
5. Select **Synchronize changes with nodes**, and then click **Save**.
6. Restart all members of the core group.

After the servers restart, all of the core group members run with the new discovery protocol setting.

Configuring core group socket buffers

Most operating systems provide program interfaces for performing operations involving the sending and receiving of data over sockets. Most operating systems also provide administrative capabilities to control the amount of memory allocated per socket that is used as data buffers.

- Check your operating system settings that are relevant to TCP sockets. For example, if you are using an AIX operating system, check the values that are specified for the tcp_sendspace, tcp_recvspace, and sb_max settings. Similarly on a Linux operating system, check the values that are specified for the tcp_rmem, and tcp_wmem settings.
- Use the WebSphere Application Server performance monitoring infrastructure to determine the average message size that the core group transport handles. If your operating system setting for the default buffer size is smaller than the average message size, make one of the following changes:
  - Change the default buffer size setting for your operating system. However, this action might be inappropriate because it might affect the operation of other applications running on this operating system.
  - Change the size of the socket buffer that the core group transport obtains. The value that is specified for the IBM_CS_SOCKET_BUFFER_SIZE core group custom property determines the size of the socket buffer that the core group transport obtains. The following table shows the relationship between the values that can be specified for this property and the underlying memory allocation size per socket buffer type:

Socket Buffer Type	Property set to No over rides	Property set to Small	Property set to Medium	Property set to Large
Unicast receive	Operating system default buffer size is used.	Buffer size is 64 kilobytes	Buffer size is 256 kilobytes	Buffer size is 1 megabyte
Unicast send	Operating system default buffer size is used.	Operating system default buffer size is used.	Buffer size is 64 kilobytes	Buffer size is 128 kilobytes

Socket Buffer Type	Property set to No over rides	Property set to Small	Property set to Medium	Property set to Large
Multicast receive	Operating system default buffer size is used.	Buffer size is 512 kilobytes	Buffer size is 1 megabyte	Buffer size is 3 megabytes

You might want to change the size of your core group buffers in the following circumstances:

- You are directed to do so by IBM Support
- You are directed to do so during the course of installing another WebSphere product.
- You want to change the behavior of the core group transport without affecting the behavior of other sockets.
- You are trying to tune the network communication path of your system to your application.

To change the socket buffer space that the core group transport allocates:

1. In the administrative console, click **Servers > Core groups > Core group settings >** *core_group_name*.
2. Under Additional Properties, click **Custom Properties**.
3. Change the value that is specified for the IBM_CS_SOCKET_BUFFER_SIZE custom property.

   If the IBM_CS_SOCKET_BUFFER_SIZE property already exists, click the property name and specify either `No over rides`, `small`, `medium`, or `large`.

   If this property does not already exist, click **New** and create it:

   a. In the Name field, specify I`BM_CS_SOCKET_BUFFER_SIZE`.

   b. In the Value field specify one of the following strings:
      - `No over rides`
      - `small`
      - `medium`
      - `large`
4. Click **OK** and then click**Review**.
5. Select **Synchronize changes with nodes**, and then click **Save**.
6. Restart all members of the core group.

After the servers restart, the core group members all run with the new socket buffer size settings.

Specifying a core group when adding a node

By default, a cell contains a single core group. In this situation, during node federation, a node agent is created and automatically added to this core group. However, if the cell contains multiple core groups, you must specify the core group to which the node agent is assigned.

If the cell to which you are adding a node agent contains multiple core groups, determine the core group to which you want this node agent to belong.

You have a cell that contains multiple core groups and you want to select the core group to which a newly created node agent is added.

To add a newly created node agent to a specific core group:

Include the coregroupname parameter on the **addNode** command. For example:

```
addNode dmgr_host dmgr_port  -coregroupname existing_core_group_name
```

The node agent resides in the specified core group.

Specifying a core group when creating an application server

By default, a cell contains a single core group. In this situation, whenever you add an application server to that cell, it is automatically added to this core group. However if the cell contains multiple core groups, you must specify the core group to which you want the application server assigned.

If the cell to which you are adding an application server contains multiple core groups, determine the core group to which you want this application server to belong.

You have a cell that contains multiple core groups and you want to select the core group to which a newly created application server is added.

To add a newly created node agent to a specific core group:

1. In the administrative console, click **Servers > Application servers > New** .
2. Select the node on which you want to create the application server.
3. Specify a name for the new application server, and then click **Next**.
4. Select the server template that you want for this application server, and then click **Next**. Usually, you want to use the template for the default server.
5. Select the core group to which you want this application server to belong from the list of available core groups, and then click **Next**.
6. Confirm the selections on the summary presented, and click **Finish**.
7. Click **OK**, and then click **Save** to save your changes.
8. Select **Synchronize changes with nodes** and click **Save** again.
9. Restart all members of the core group.
10. Start the server.

After the server starts, it becomes a member of the selected core group.

Viewing the core groups in a cell

A core group is a component of the high availability manager. A default core group, called DefaultCoreGroup, is created for each cell. A core group can contain application servers, proxy servers, node agents, and the deployment manager. A core group must contain at least one node agent or the deployment manager.

If you need to move servers between core groups, in preparation for the move, you can view the list of the different core groups contained in a cell to help determine to which core group you want to move a server.

To view the core groups that are in a cell:

In the administrative console, click **Servers** > **Core groups** > **Core group settings**.

A list of the core groups that are in the cell is displayed.

Click the name of one of the core groups to obtain specific information about that core group.

Core group collection

A core group is a component of the high availability manager function. A default core group, called DefaultCoreGroup, is created for each cell in the WebSphere Application Server environment. A core group can contain standalone servers, cluster members, node agents and the deployment manager. A core group must contain at least one node agent or the deployment manager.

To view this administrative console page, click **Servers > Core Groups > Core group settings** .

Click **New** to define a new core group. After a core group is defined, several fields become read-only. To change those fields, delete and redefine the core group.

Click **Delete** to delete a core group. A core group must be empty before it can be deleted.

Name

Specifies the name of the core group. Click the core group name to edit the settings for that core group. This field is read-only.

Description

Specifies a description of the core group. This field is read-only.

Connected core groups

Specifies the core groups that are connected to this core group by access points. This field is read-only.

Viewing core group members

A core group member is an application server, proxy server, the deployment manager, or a node agent that is a member of a high availability core group.

If you need to move servers between core groups, in preparation for the move, you can view the list of servers that belong to each core group to help determine which servers you want to move to a different core group.

To view the members of a core group:

1. In the administrative console, click **Servers > Core groups > Core group settings**. A list of core groups that are in the cell is displayed.
2. Click the name of one of the core groups and then click **Core group servers**.

A list of the core group members is displayed.

In the administrative console, you can click **Servers > Core groups > Core group settings > Preferred coordinator servers** to determine which of the core group members are on the list of preferred coordinator servers for this core group. You should remove any servers that you intend to move to a different core group from this list before changing their core group association.

Core group servers collection

Use this page to view the servers that are part of a core group. A core group server can be an application server, a deployment manager, or a node agent that is a member of a high availability core group. Use this page to move servers into a different core group. All members of a cluster must be in the same core group. If you select one or more members of a cluster, all of the members of that cluster must be moved.

To view this administrative console page, click **Servers > Core groups > Core group settings > New** or select an existing core group for editing. Then click **Core group servers**.

Name

Specifies the names of the servers in the core group. This field is read-only. Click this field to specify custom properties for this server.

Node

Specifies the node that contains the core group server. This field is read-only.

Version

Specifies the product version of the node in the cell. A Version 6 deployment manager node can own a Version 5 managed node. Version 5 managed nodes are easily identified in this field. This field is read-only.

Type
Specifies the server process type, which can be either deployment manager, node agent, or application server. Standalone application servers in the cell, managed nodes, and cluster members all display as application server types. This field is read-only.

Cluster Name
Specifies the cluster name if the core group server is part of a cluster. If the core group server does not belong to a cluster, this field is blank. This field is read-only.

To move one or more servers to another core group, select the check box next to the names of the servers that you want to move and click **Move**.

Important: You must stop a core group server before you move it to another core group.

Core group server settings

Use this page to create or change custom properties for a server in a core group.

To view this administrative console page, click **Servers** > **Core groups** > **Core group settings** > ***Select an existing core group*** > **Core group servers** > ***Select an existing server***.

Extended information about the fields on the panel:

Node
Specifies the name of the node that contains the core group server. This field is read-only.

Name
Specifies the name of the core group server. This field is read-only.

Custom Properties
Click on this field to define or edit a custom property for the core group server.

Creating a new core group

A default core group, called DefaultCoreGroup, is created for each cell. This default core group is sufficient in most configurations. However there are some circumstances under which you need to create additional core groups for a cell.

Determine how to segment your existing cell into multiple core groups. Use the following rules as guidelines:

- All members of a cluster must be in the same core group. A core group can contain multiple clusters.
- Core group members cannot cross firewall boundaries.
- Put clusters with direct relationships in the same core group. Examples of direct relationships include:
  - A single application is deployed on multiple clusters.
  - An application on one cluster calls an application on another cluster.
- Each core group must contain at least one deployment manager or node agent process.

You might want to add another core group to a cell under the following circumstances:

- A firewall separates members of an existing core group, for example a proxy server and an application server are separated by a firewall.
- You are scaling up the number of servers in a cell. Multiple core groups are recommended for a large cell.

To create a new core group:

1. In the administrative console, click **Servers** > **Core groups** > **Core group settings** > **New** .

2. In the Name field, specify a unique name for the new core group. The name can contain alpha and numeric characters, but not the following special characters:

   ```
   # \ / , : ; " * ? < > | = + & % '
   ```

   The name also cannot begin with a period (.) or a blank space. A blank space does not generate an error. However, leading and trailing blank spaces are automatically deleted from the name.
3. Add a description of this core group that helps other administrators understand the purpose of this core group.
4. Click **OK** and then click**Review**.
5. Select **Synchronize changes with nodes**, and then click **Save**.

The cell contains another core group.

You must now:

- Complete your core group configuration. The initial core group settings and policies are derived from a template. If the settings from the default template do not meet your requirements, you can:
  - Change the number of coordinators for this core group.
  - Change the transport type for this core group.
  - Add policies for this core group.
- Move members to the new core group.
- Create bridges between core groups. If clusters with direct relationships are not in the same core group, set up a core group bridge to connect the related core groups.

.

Moving core group members

When moving members to a different core group, remember that: each WebSphere process is a member of exactly one core group, all members of a given cluster must belong to the same core group, and each core group must contain at least one deployment manager or node agent process.

- Review core group concepts.
- Determine which core group members you want to move, and to which core group you want to move them
- Stop affected processes using the following guidelines:
  - Case 1: You are not moving the deployment manager. Stop all of the processes you are moving.
  - Case 2: You are moving the deployment manager. Because the deployment manager has to be moved separately, stop all of the other processes you are moving, but leave the deployment manager running until after you have moved all of these other processes.

    Important: In general, you should not move a deployment manager.

You might need to move one or more core group members:

- To populate a newly created core group.
- To rebalance existing core groups.

To move members between core groups:

1. In the administrative console, click **Servers** > **Core groups** > **Core group settings** > *core_group_name* > **Core group servers**.
2. Select the core group containing the processes that you want to move to another core group.
3. In the Select column, select the servers that you want to move. If you are populating a new core group, at least one node agent or deployment manager must be moved to the new core group.

4. Click **Move**. The **Core groups > DefaultCoreGroup > Core group servers > Move** administrative console panel is displayed showing the servers you want to move and the core group to which these servers currently belong.
5. Indicate in the To core group field the core group to which you want these servers moved.
6. Click **Apply** and then **Save**.
7. Click **System administration > Nodes**, and then click **Synchronize** to synchronize your changes on all running nodes.
8. Manually synchronize all stopped nodes by running the **syncNode** command from the *profile_root*/*node_agent_profile*/bin directory.
9. If the deployment manager is moved, restart the deployment manager process.
10. Restart all of the other moved processes.

After the servers complete their restart, all moved servers should belong to their new core group.

- You can verify that the servers are in the correct core groups. For each core group, in the administrative console, click **Servers > Core groups > Core group settings >** *core_group_name* **> Core group servers** and look at the list of core group servers that displays.
- You can set up core group bridges if any of the core groups need to communicate with each other. See "Core group communications using the core group bridge service" on page 495 for more information.

Core group server move options

Use this page to move one or more core group servers to a different core group. You must stop a core group server before you move it.

Servers can be moved from one core group to another, as long as the following core group requirements are not violated:

- A non-empty core group retains at least one node agent or deployment manager as a member of that group. (The high availability manager configuration change listeners are only available on the node agent or deployment manager servers.
- All members of a cluster must be members of the same core group. If one or more of the servers you are moving belongs to a cluster, you must move all of the members of that cluster. (A core group can span multiple WebSphere clusters.)

To view this administrative console page, click **Servers > Core groups > Core group settings >** *core_group* **> Core group servers**. Select the servers to be moved and then click **Move**.

Extended information about the core group fields:

Move selected servers	Specifies the servers that you selected to be moved. It cannot be edited.
From core group	Specifies the name of the core group that you are moving the servers from. It can not be edited.
To core group	Specifies the core group to which these servers will belong.

Click **Apply** to make your changes effective. Click **Save** and save and synchronize your changes with all managed nodes.

Disabling or enabling a high availability manager

A unique HAManagerService configuration object exists for every core group member. The enable attribute in this configuration object determines if the high availability manager is enabled or disabled for the corresponding process. When the enable attribute is set to `true`, the high availability manager is enabled. When the enable attribute is set to `false`, the high availability manager is disabled. By default, the high availability manager is enabled. If the setting for the enable attribute is changed, the corresponding process must be restarted before the change goes into effect. You must use the wsadmin tool to disable or enable a high availability manager.

- Determine if you need to use a high availability manager to manage members of a core group.
- Add a script file that is similar to the following file to your system. This sample script file disables the high availability manager on a specific process. You can modify this script file if you need to disable the high availability manager on all of the processes in a cell or on all of the processes that are members of a specific core group. You can also modify this script file to enable a high availability manager that you previously disabled.

```
#####################################################################
# Script name = disableHamOnProcess.pty
#####################################################################

def getHAMServiceOnAll():
    # get a list of all HAManagerService objects in the cell.
    processes = AdminConfig.list("HAManagerService").split("\n")
    rc = []
    for p in processes:
        p = p.strip()
        rc.append(p)
    return rc

# The HAManagerService ObjectName has the following format
#cells/cellname/nodes/nodename/servers/servname:hamanagerservice.xml
#id def getNodeName(service):
    # The 4th /-separated element in the service name is the node name
    n = service.split("/")[3]
    return n

def getProcessName(service):
    # The 6th /-separated element in the service name is the process name
    p = service.split("/")[5]
    return p.split("|")[0]

def printHelp():
    print "This script disables the HA Manager on a specific process"
    print "Format is disableHamOnProcess nodeName processName"

##################################
# main
#################################
if(len(sys.argv) > 1):
    # get node name and process name from the command line
    nodeName = sys.argv[0]
    processName = sys.argv[1]
    # get a list of all HAManagerService objects in the cell.
    processes = getHAMServiceOnAll()
    for p in processes:
        # debug
        print "Checking process "+p
        # Check for a node name match.
        n = getNodeName(p)
        if (nodeName == n):
            # node name matches, check for server name match
            pn = getProcessName(p)
            if (pn == processName):
                # both node and process names match. Found the one we
                # are looking for. Disable and exit.
                print "Disabling the HA Manager on process ",
                print p
                AdminConfig.modify(p, [["enable", "false"]])
                AdminConfig.save()
                break
else:
    printHelp()
```

You might want to disable a high availability manager if you are trying to reduce the amount of resources, such as CPU and memory, that WebSphere Application Server uses and have determined that the high availability manager is not required on some or all of the processes in a core group.

You might need to enable a high availability manager that you previously disabled because you are installing applications on core group members that must be highly available.

You must use the wsadmin tool to disable a high availability manager or to enable a high availability manager that you previously disabled.

1. Start the wsadmin tool.
2. Start the deployment manager for the cell that contains the high availability manager that you are disabling or enabling. The deployment manager for this cell must be running for the sample script to work.
3. Use the wsadmin tool to issue the following command to start the script:

   ```
   -lang jython -f script_file_name node_name process_name
   ```

   For example, to start the sample script for the WASDOCLNodePH02 node and the ClusterMember2 process, issue the following command:

   ```
   -lang jython -f disableHamOnProcess.pty WASDOCLNodePH02 ClusterMember2
   ```
4. Restart all processes with changed setting for the enable attribute.

The processes start with the high availability manager in the appropriate state.

Viewing high availability group information

High availability groups are dynamically created components of a core group. They cannot be configured directly, but they are directly affected by static data, such as policy configurations, which is specified at the core group level. You can use the administrative console to view information about the high availability groups that are part of a core group.

You need to know the name of the core group that contains the high availability groups you want to view. You should also determine if you need to view all of the high availability groups or a subset of these groups, based on the group name.

You might want to perform this task if:

- You want to view the current set of high availability groups.
- You want to view the group name of a high availability group.
- You want to view the policy that is associated with a high availability group.
- You want to view the state of a high availability group

To view information about the high availability groups contained in a core group:

1. In the administrative console, click **Servers > Core groups > Core group settings**.
2. Click the core group that contains the high availability groups you want to view.
3. Click the **Runtime** tab.
4. Specify a value in the **Group name** field.
   - Specify an asterisk in this field if you want a list of all the high availability groups that are part of this core group.

- Specify a set of name-value pairs, separated by a comma, to get a list of only those high availability groups that contain the specified name-value pairs in their group name. For example, you might specify the following value to obtain a list of all of the high availability groups that contain `IBM_hc=MyCluster` and `type=WAS_TRANSACTIONS` in their names.

  ```
  IBM_hc=MyCluster,type=WAS_TRANSACTIONS
  ```

5. Click **Show groups**.

A list of high availability groups that are contained in this core group that meet the specified criteria is displayed, with pertinent information about these groups.

You can click on the name of one of the high availability group names to display information about members of that group. Because high availability group members are dynamically created, no member information is available for configured servers that are not actually running.

You can also view the distribution of active high availability group members.

Viewing the distribution of active high availability group members

Because high availability group members are dynamically created, core group policy configuration is used to determine which high availability group members the high availability manager activates. In some situations, it is possible for active members of multiple high availability groups to all be running on the same server process. You can use the administrative console to view the distribution of active high availability group members across your application servers.

You need to know the name of the core group that contains the high availability groups you want to view. You should also determine if you need to view all of the high availability groups or a subset of these groups, based on the group name.

You might want to perform this task if:

- You want to view the current active high availability group member distribution for the servers in a core group.
- You want to determine whether a particular server is overloaded because it is hosting the active member for multiple high availability groups.
- You want to check the current active member distribution before you update policies that might affect this distribution.
- You want to verify that you obtained the proper results to the current active member distribution after a policy change goes into affect.

To view the distribution of active high availability group members:

1. In the administrative console, click **Servers > Core groups > Core group settings**.
2. Click the core group that contains the high availability groups you want to view.
3. Click the **Runtime** tab.
4. Specify a value in the **Group name** field.
   - Specify an asterisk in this field if you want a list of all the high availability groups that are part of this core group.
   - Specify a set of name-value pairs, separated by a comma, to get a list of only those high availability groups that contain the specified name-value pairs in their group name. For example, you might specify the following value to obtain a list of all of the high availability groups that contain `IBM_hc=MyCluster` and `type=WAS_TRANSACTIONS` in their names.

     ```
     IBM_hc=MyCluster,type=WAS_TRANSACTIONS
     ```

5. Click **Show servers**.

A list of servers displays that shows the number of active high availability group members on each server.

Servers with active members collection

Use this page to determine how many high availability group members are active on a particular application server.

To view this administrative console page, click **Servers > Core groups > Core group settings >** *core_group*. Click on the **Runtime** tab and specify group name properties for a high availability group. (You can specify an asterisk (*) to get a list of the servers that are hosting active members for all the high availability groups in this core group.) Then select **Show severs**.

Server

Specifies the name of a server on which there are active high availability group members. This field is read-only.

Node

Specifies the node on which each server is running. This field is read-only.

Version

Specifies the version of the WebSphere Application product on which each node is running. This field is read-only.

Active members

Specifies the number of high availability group members that are currently active on that server. This field is read-only.

High availability groups collection

Use this page to view information about the high availability groups contained in a core group.

To view this administrative console page, click **Servers > Core groups > Core group settings >** *core_group*. Click on the **Runtime** tab. In the Group name properties field, specify a match criterion for a specific high availability group, or specify an asterisk to get a list of all the high availability groups that are part of this core group. (A match criterion is one or more name=value pairs that match attributes contained in a high availability group's name.) Then click **Show groups**.

High availability group
Specifies the names of the high availability groups. The name of a high availability group is a set of name-value pairs or attributes, separated with commas. For example, `name=productiongroup,policy=abc,ibm=websphere` could be the name of a high availability group. This field is read-only.

Quorum
Specifies if quorum is enabled for each high availability group. This field is read-only.

Policy

Specifies the policies that have match criteria that matches properties contained in the name of that high availability group. There should only be one policy listed for a high availability group. However, if multiple policies have match criteria that equally match properties in a high availability group's name, all of the policies with matching criteria are listed, and the ERROR icon appears in the status column.

For example, if you have a high availability group named `name=productiongroup,policy=abc,ibm=websphere`, and MyPolicy1 has the match criteria `name=productiongroup`, and MyPolicy2 has the match criteria `policy=abc`, both MyPolicy1 and MyPolicy2 are considered matching policies and are listed in the Policy column.

This field is read-only.

Status

Specifies, with icons, whether or not only one policy is associated with a high availability group. If the OK icon displays in this column, a single policy is associated with that high availability group. If the ERROR icon displays in this column, multiple policies are associated with that group.

If the ERROR icon displays for a high availability group, you must adjust the match criteria for one or more of the policies listed in the Policy column for that group so that the correct policy is the only one associated with that high availability group.

The match criteria for multiple policies can match some of the same properties in a group's name as long as one policy has a match criteria that matches more of the properties in that group's name than the match criteria of any of the other policies. For example, if you have a high availability group with a name that consists of the following name and value pairs:

```
name=productiongroup,policy=abc,ibm=websphere
```

and MyPolicy1 has the match criteria `name=productiongroup` and MyPolicy2 has the match criteria `name=productiongroup,ibm=websphere`, MyPolicy2 is considered the matching policy because it has more match criteria that matched the properties contained in the name of the high availability group.

This field is read-only.

Use the **Disable** button to disable all of the members of a high availability group that were previously active or idle. One of the few times you might want to use this button is if you are planning to remove or delete all of the servers on which this group has a member running.

Use the **Enable** button to enable all of the members of a high availability group that were previously disabled. These members can then be activated according to the policy associated with that group.

High availability group members collection

Use this page to view information about the individual members of a high availability group. This page lists the current members of the selected high availability group.

To view this administrative console page, click **Servers > Core groups > Core group settings >** *core_group*. Click on the **Runtime** tab. In the Group name properties field, specify a match criterion for a specific high availability group, or specify an asterisk (*) to get a list of all the high availability groups that are part of this core group. (A match criterion is one or more name-value pairs that match attributes contained in the name of a high availability group.) Then click **Show groups** and select one of the high availability groups listed.

Name
Specifies the name of a high availability group member.

Node
Specifies the node on which each high availability group member is running.

Version
Specifies the version of the WebSphere Application Server product on which each node is running.

Status
Specifies the state of the high availability group members.

High availability group members are either idle, activated, or disabled. The usual states are idle or activated. One of the few times you might want to disable a member is if it is running on a server that you plan to remove or delete.

- If a group member is idle, it cannot be assigned any work.
- If a group member is activated, it can be assigned work.
- If a group member is disabled, it must be enabled before it can be activated.

Click the **Disable** button to prevent a group member from participation in the group. A member in the disabled state can never be made active or used by the group.

Click the **Enable** button to enable a group member that was previously disabled.

Click the **Activate** button to activate an idle group member.

Click the **Deactivate** button to make an active group member idle.

Creating a policy for a high availability group

Every high availability group has to have an associated policy. This policy determines which members of a high availability group to put in the active state.

Before creating a new policy, you should review the following topics:

-
- “High availability group policy modification guidelines” on page 459
- “High availability group policies” on page 453

You should also know:

- The name of the core group that you want to associate with the new policy.
- The name of the high availability group that you want this policy to control.
- The function, such as transaction log recovery or messaging engine, that is associated with this high availability group.
- The policy types, such as One of N or Static, that this function supports.
- The type of policy you want to create.
- The policy settings, such as failback, and preferred servers only, that you want to configure for this policy.

WebSphere Application Server includes default policies that are already associated with the high availability groups some of the WebSphere Application Server components use. If these default policies do not meet the requirements of your installation, it is recommended that you create a new policy instead of changing one of the default policies. The creation of new policies provides you with the capability to tailor the policy settings to your installations requirements while giving you the option to revert back to the default policy.

To create a new policy:

1. In the administrative console, click **Servers > Core groups > Core group settings** *core_group_name* **> Policies > New**.
2. Select the new policy you want in effect for a specific high availability group. If you need to define a new policy, the policy options are:
   - All active policy: All of the group members are activated.
   - M of N policy: *M* group members are activated. The number that is represented by *M* is defined as part of the policy details.
   - No operation policy: No group members are activated.
   - One of N policy: Only one group member is activated.
   - Static policy: The active members of a group are statically configured.
3. Click **Next**.
4. Specify a name for the policy in the Name field. The name must be unique within the scope of the core group. Make the name meaningful to other administrators.
5. **Optional:** Specify a Description of the policy in the description field. This description might include the name of the associated core group.
6. Specify a value for the Is alive timer field, if the default value is too long or too short a time period. This value determines how frequently the high availability manager checks the health of the high availability group members. The default value is 0 seconds.
   - If you specify -1 (minus 1), the Is alive timer is disabled.
   - If you specify 0 (zero), the value that is specified for the Is alive timer at the core group services level is used for high availability groups that are associated with this policy.
   - If you specify an integer between 1 and 2147483647, inclusive, this value is used for the high availability groups that are associated with this policy.
7. Make sure the Quorum field is not selected. You should not enable Quorum unless you are explicitly instructed to do so in the documentation for some other product.
8. Select the **Failback** field if you want to have the high availability manager make the most preferred member the currently active member whenever this action is possible. This option is available only for M of N and One of N policies.
9. Select the **Preferred servers only** field if you want the high availability manager to only activate group members on servers that are contained in the Preferred servers list. This option is available only for M of N and One of N policies. If you select this option, you must configure a list of preferred servers. A description of how to set up this list is provided in a later optional step.
10. Specify the number of group members that you want active in the Number of active members field. This option is available only for an M of N policy.
11. Click **Apply** and then select **Match criteria**.
12. On the next panel, click **New** and then configure the match criterion for this policy.
    a. In the Name field, specify the name of one of the name-value pairs contained in the name of the high availability group that you want to associate with this policy.
    b. In the Value field, specify the value of the name-value pair you specified in the Name field.
    c. **Optional:** In the Description field, add a description of this match criterion . For example, you might specify First attribute to indicate that this name-value pair matches the first attribute contained in the group name.

d. Click **OK**.

e. Repeat these steps for each additional attribute you want to include as part of your match criterion.

You should set the match criterion for a new policy to two or more of the name-value attributes that are contained in the name of the high availability group to ensure that this policy is used instead of one of the WebSphere Application Server default policies. Using this example, the following high availability group-to-policy association is established:

13. Uunder **Additional Properties**, select the **Static group servers** field to configure the list of servers that you want activated. This option is available only for Static policies. Click **Add** to move core group servers into the list of Static group servers, and then Click **OK** after you complete the list.
14. **Optional:** Under Additional Properties, select **Preferred servers** and select the preferred servers for this policy. This option is available only if you selected the Preferred servers only field for M of N and One of N policies. If you do not set up this list, no group members are activated.

    Click **Add** to move core group servers into the list of preferred servers.

    Select specific servers in the list and click **Move up** and **Move down** to adjust the order of the servers within the list. Make sure that the most preferred server is at the beginning of the list and the least preferred server is at the end of the list.

    After you complete the preferred servers list, click **OK**.

    Important: Use caution when selecting preferred servers. WebSphere Application Sever cannot detect if you select an inappropriate server as a preferred server. For example, if the policy affects a messaging engine or transaction service, only select preferred servers from the messaging engine cluster. Similarly, if the policy affects a transaction service, only select preferred servers from the transaction service cluster.
15. Click **OK** and then click**Review**.
16. Select **Synchronize changes with nodes**, and then click **Save**.

The new policy goes into affect after it is saved and synchronized. You do not have to stop and restart the affected application servers.

You can change the **Failback** and **Preferred servers only** options for this policy without stopping and restarting the affected application servers.

You can create or update the list of preferred servers that for this policy without stopping and restarting the affected application servers.

Core group policies

Use this page to create or update the various high availability group policies. For a given high availability group, the associated policy determines which members of the group should be made active.

To view this administrative console page, click **Servers > Core groups > Core group settings > New** or *existing core group* > **Policies**.

Click **New** to define a new policy. After a policy is defined there are several fields that you can no longer change. To change those fields, delete and redefine the policy. Click **Delete** after selecting a policy to delete the selected policy.

After adding a high availability group, you need to take more actions to enable workload balancing for messaging resources. For more information about the extra actions, see the Related tasks.

All of the policy fields on this page are read-only. To change the values specified in any of these fields, click on the name of the policy you want to change. When the console page **Core group settings** > *group_name* > **Policies** > *policy name* displays, you can edit the policy properties.

Name
Specifies the name of the policy.

Description
Specifies a description of the policy.

Policy type
Specifies the desired policy type.

Restrictions:

1. If you are setting up a policy for a transaction manager, you must select One of N as the policy type.
2. If you are setting up a policy for a service integration bus you must select One of N or Static as the policy type. The default policy that IBM provides for a service integration bus uses a One of N policy type.

Following is a list of valid policy types:

All active
The All active policy indicates that the high availability manager keeps all of the application components that are running on all of the servers in the high availability group active at all times.

M of N
The M of N policy is similar to the One of N policy. However, it enables you to specify the number (M) of high availability group members that you want to keep active if it is possible to do so. The number of active members must be greater than one and less than or equal to the number of servers in the high availability group. If the number of active servers is set to one, this policy is a match for the One of N policy.

No operation
The No operation policy indicates that no high availability group members are made active.

One of N
The One of N policy keeps one member of the high availability group active at all times. This is used by groups that desire singleton failover. If a failure occurs, the high availability manager starts the singleton on another server.

Static
The Static policy allows you to statically define or configure the active members of the high availability group.

Match criteria
Specifies one or more name-value pairs that are used to associate this policy with a high availability group. These pairs must match attributes that are contained in the name of a high availability group before this policy is associated with that group.

Core group policy settings

Use this page to define a policy for a high availability group. A policy is defined at the core group level. It only applies to matching high availability groups contained within this core group

To view this administrative console page, click **Servers > Core groups > Core group settings > New** or *existing core group* > **Policies** > *policy_name*.

Name

Specifies the name of the policy. This name must be unique within the scope of a core group.

Policy type

Specifies the policy type that was selected when this policy was created. This is a read-only field. If you want to change the policy type, you must delete this policy and then create it again specifying a different policy type. If this is an IBM provided policy, do not delete it. Instead create a new policy and specify more of the attributes contained in the name of the high availability group as the match criterion for this new policy. The policy with the greatest number of matches to attributes in a group's name is the policy that is associated with that group.

Description

Specifies a description of this policy. For example, the clustered TM policy provided with the product has ″TM One-Of-N Policy″ as its description.

Is alive timer

Specifies, in seconds, the interval of time at which the high availability manager will check the health of the active group members that are governed by this policy. If a group member has failed, the server on which the group member resides is restarted.

The high availability manager detects two fundamentally different kinds of failures.

- An entire process failure. This failure detection is accomplished using functions such as the heartbeat timers. This type of detection does not involve the Is alive timer function. If an entire process fails, it will be detected and the various high availability groups will fail over to other servers in the core group.
- An application or program failure. If, for some reason. an application or a program, like the transaction manager or a service integration bus function hangs, the high availability manager will eventually detect the hang. The amount of time that might pass before the hang is detected is determined by the value specified for the Is alive timer parameter. The parameter controls how often the high availability manager will call back to the component that created the high availability group member and ask if it is still alive. This allows detection of hung code or program errors that somehow do not cause the entire process to stop functioning or terminate.

Data type	Integer • Valid values are -1 to 600 seconds, inclusive. • If -1 (negative 1) is specified, this function is disabled. • If 0 (zero) is specified, the frequency at which the high availability manager checks the health of the active group members is determined by the time interval specified at the application server process level. • If a value larger than 0 (zero) is specified, the high availability manager uses the time interval specified here, instead of the one specified at the application server process level, when determining how frequently it should check the health of the high availability group members using this policy.
Default	0 (zero)

Quorum

Specifies whether quorum checking is enabled for a group governed by this policy. Quorum is a mechanism that can be used to protect resources that are shared across members of the group in the event of a failure.

Important: Quorum is an advanced hardware function and should not be enabled unless you thoroughly understand how to properly use this function. If not used properly, this function can cause data corruption.

The Quorum setting in the policy will only have an effect if the following items are true:

- The group members are also cluster members.
- GroupName.WAS_CLUSTER=*clustername* must be specified as a property in the group name of any high availability group matching this policy.

When enabled, any group using this policy will not achieve quorum until a majority of the members are running. For example, if there are n members in the group, (n/2) + 1 servers must be online in order to achieve quorum. No group members will be activated until quorum has been achieved.

The quorum mechanism is designed to work in conjunction with a hardware control facility that allows application servers to be shut down if a failure causes the group to be partitioned.

Fail back

Specifies whether work items assigned to the failing server are moved to the server that is designated as the most preferred server for the group if a failure occurs. This field only applies for M of N and One of N policies.

Preferred servers only

Specifies whether group members are only activated on servers that are on the list of preferred servers for this group. This field only applies for M of N and One of N policies.

Number of active members

Specify how many of the high availability group members are to be activated. This field only applies for the M of N policy.

Additional Properties

Specifies one or more of the following options, depending on the type of policy you selected:

Custom properties	Click to specify custom properties for the policy.
Match criteria	Click to set up a match criterion for the policy.
Preferred servers	Click to set up a list of servers that are given preference when group members are activated.
Static group servers	Click to set up a list of the specific servers that are activated.

New core group policy definition

Use this page to create a new policy for a high availability group.

When you create a new policy, the first page that displays lets you select a policy type. To view the administrative console page where you select a policy type, click **Servers > Core groups > Core group settings > New** or select an existing core group. Then click **Policies > New**.

Select one of the following policy types:

- All active policy: Under this policy, all of the group members are activated.
- M of N policy: Under this policy, *M* group members are activated. The number represented by *M* is defined as part of the policy details.
- No operation policy: Under this policy, no group members are activated.
- One of N policy: Under this policy, only one group member is activated.
- Static policy: Under this policy, only specified group members are activated.

After selecting a policy, click **Next** to continue.

Preferred servers

Use this page to define the ordered list of *preferred servers* for the selected policy. The policy gives preference to the servers in this list when activating group members.

To view this administrative console page, you must be working with a policy that has a policy type of M of N or One of N. If your policy has one of these policy types, click **Servers > Core groups > Core group settings > New >** or *select an existing core group* **> Policies > New >** or *select an existing policy*. Under **Additional Properties**, select **Preferred Servers**.

Use **Add** and **Remove** to move servers into and out of the list of preferred servers. Use **Move up** and **Move down** to adjust the order within the list of preferred servers. Make sure that the most preferred server is at the top of the list and the least preferred server is at the bottom.

Click **OK** to make your changes effective. Click **Save** to save and synchronize your changes with all managed nodes.

Changes to the preferred servers list take affect as soon as they are saved and synchronized. You do not have to stop and restart the affected application servers.

Match criteria collection

Use this page to view the match criteria that are defined for a policy.

To view this administrative console page, click **Servers** > **Core groups** > **Core group settings** > **New** > or *select an existing core group* > **Policies** > **New** or *select an existing policy* > **Match criteria**.

Click **New** to create a new match criterion for the policy. Click the name of a match criterion to change any of that criterion’s properties.

Name

Specifies the name portion of a name-value pair that is part of the name of the high availability group that you are associating with this policy.

Value

Specifies the value portion of a name-value pair that is part of the name of the high availability group that you are associating with this policy.

Description

Specifies a description of the match criterion. Make the description meaningful. For example, the description might indicate the high availability group that this name-value pair matches.

Match criteria settings

Use this page to define a match criterion for a policy.

To view this administrative console page, click **Servers** > **Core groups** > **Core group settings** > **New** > or *select an existing core group* > **Policies** > **New** or *select an existing policy* > **Match Criteria** > *criterion name*.

The name and value fields should match a name-value attribute included in the name of a high availability group you want associated with this policy.

After you define a match criterion, click **Apply** to make your changes effective. Click **Save** to save and synchronize your changes with all managed nodes.

Name

Specifies the name portion of a name-value pair that is part of the name of the high availability group that you are associating with this policy.

Value

Specifies the value portion of a name-value pair that is part of the name of the high availability group that you are associating with this policy.

Description

Specifies a description of the match criterion. Make the description meaningful. For example, the description might indicate the high availability group that this name-value pair matches.

Static group servers collection

Use this page to designate for a static policy which high availability group members should be made active.

This option is available under Additional Properties only if Static is selected as the policy type. To view this administrative console page, click **Servers > Core groups > Core group settings >** *existing_core_group* **> Policies >** *static_policy_name* > **Static group servers**.

Use **Add** and **Remove** to move servers into and out of the list of servers that should be activated. Only high availability group members that are associated with this policy appear on this page.

After you finish updating the list, click **Apply** to make your changes effective. Click **Save** to save and synchronize your changes with all managed nodes.

Selecting the policy for a high availability group

Every high availability group has an associated policy. The high availability manager uses this policy to determine which members of a high availability group to put in the active state.

Before you select a policy for a high availability group, you should review the following topics:

-
- "High availability group policy modification guidelines" on page 459
- "High availability group policies" on page 453

You should also know:

- The name of the core group that you want to associate with the new policy.
- The name of the high availability group that you want this policy to control.
- The function, such as transaction log recovery or messaging engine, that is associated with this high availability group.
- The policy types, such as One of N or Static, that this function supports.
- The type of policy you want to create.
- The policy settings, such as failback, and preferred servers only, that you want to configure for this policy.

You have multiple policies defined for a high availability group, and you want to specify which of these policies the high availability manager uses to govern the group.

To select a policy for a high availability group:

1. In the administrative console, click **Servers > Core groups > Core group settings** *core_group_name* .

2. Click the **Runtime** tab to determine both the name of the high availability group, and the name of the policy that is currently controlling the group. See “Viewing high availability group information” on page 482 for more information on how to perform this step. You must have at least one of the group members running.
3. Click the **Configuration** tab to determine the match criteria defined in the current high availability group policy.
4. Use the information you obtained in the previous steps and update the match criteria for the policy you are selecting. The match criterion must contain all of the match criteria from the original policy, and at least one additional attribute from the name of the high availability group.
5. Click **OK** and then click**Review**.
6. Select **Synchronize changes with nodes**, and then click **Save**.

The high availability manager uses the new policy to govern the designated high availability groups.

Specifying a preferred server for messaging requests

If a core group includes a cluster of application servers, and a messaging engine is configured for that cluster, any of the servers in that cluster can handle work items for the messaging engine. The default message provider in WebSphere Application Server is based on Service Integration Bus (SIB) technology, and is governed by the Default SIBus policy, which is a One of N policy. This policy ensures that only one of the application servers in the cluster is active at a time). You can modify the high availability group policy to specify that a specific cluster member handles the messaging work.

Before specifying a preferred server for messaging requests:

- You should review the following topics:
  - “High availability groups” on page 452
  - “High availability group policy modification guidelines” on page 459
  - “High availability group policies” on page 453
- You must determine:
  - The name of the core group that includes the server that you want to handle messaging requests.
  - The name of the high availability group for the messaging function.
  - The name of the policy that is associated with this high availability group
- You must create a new policy specific to the high availability group that controls the messaging engine cluster, if one does not already exist.

  It is possible for a single policy to govern several different high availability groups. Therefore, to modify the policy for cluster scoped control, you must create a new policy specific to the high availability group that controls the messaging engine cluster. See “Creating a policy for a high availability group” on page 486 for more information on how to create this policy.

  After you create the new policy and associate the policy with the high availability group for a given cluster, You can specify a preferred server for messaging requestws..

For high availability, you must configure a messaging engine to run in a cluster. However, you might want a specific cluster member to handle the messaging requests. Another member of the cluster should handle the messaging requests only if the preferred member fails.

1. In the administrative console, click **Servers > Core groups > Core group settings >** *core_group_name*.
2. Click the **Runtime** tab to determine both the name of the high availability group, and the name of the policy that is currently controlling the group. See “Viewing high availability group information” on page 482 for more information on how to perform this step. You must have at least one of the group members running.

3. In the administrative console, click **Servers > Core groups > Core group settings >** *core_group_name* **> Policies**.
4. Click the name of the policy that you want to modify.
5. Under Additional Properties, select **Preferred servers** and select the preferred servers for this policy.

   Click **Add** to move core group servers into the list of preferred servers.

   Select specific servers in the list and click **Move up** and **Move down** to adjust the order of the servers within the list. Make sure that the most preferred server is at the beginning of the list and the least preferred server is at the end of the list.
6. After you complete the preferred servers list, click **OK**.
7. Click **OK** and then click**Review**.
8. Select **Synchronize changes with nodes**, and then click **Save**.

All work items for the messaging engine on the associated cluster are routed to the new preferred server.

Configuring the core group bridge service

The core group bridge service can be configured for communication between core groups. A core group is a statically defined component of the high availability manager. To configure communication between core groups, use an access point group. An access point group is a collection of core groups that communicate with each other.

Review the following topics before you configure a core group bridge service:

- “Core groups” on page 441, which describes a core group.
- “Creating a new core group” on page 478, which describes how to configure a core group.
- “Configuring communication between core groups that are in the same cell” on page 511, which describes how to configuring communication between core groups that are in the same cell.

You must configure the core group bridge service whenever two or more core groups are configured in the same cell. You can also configure the core group bridge to share traffic among core groups that are in different cells. Configure the core group bridge to communicate between cells only when the service is required by another WebSphere Application Server component. By configuring the core group bridge service, the availability status of the servers in each core group is shared among all the configured core groups. For more information, see “Core group communications using the core group bridge service.” You can configure core groups to communicate in the following ways:

- Use the core group bridge for communication between core groups that are in different cells. Configuring this type of communication is the most common core group scenario. You can configure each cell to communicate with one or more other cells. For more information, see “Configuring the core group bridge between core groups that are in different cells” on page 497.
- Use advanced configurations. You might need to configure core group communication between core groups that are in the same cell, that communicate across different networks, or that use a proxy peer. For more information, see “Creating advanced core group bridge configurations” on page 505.

Multiple core groups can communicate with each other.

Continue configuring the high availability environment. See Chapter 9, “Setting up a high availability environment,” on page 437 for more information.

Core group communications using the core group bridge service

The core group bridge service can be configured to share availability information about internal WebSphere Application Server components between core groups. For example, by configuring the core group bridge service, each core group can be aware of the status of all of the application servers that are configured in all of the core groups. Use access point groups to define the core groups that communicate. Do not use the core group bridge service to share application information among core groups.

A core group is a statically defined component of the high availability manager. Each cell must have at least one core group. WebSphere Application Server creates a default core group called **DefaultCoreGroup** for each cell. For more information about core groups, see “Core groups” on page 441. Two or more core groups can be set up to communicate with each other and share workload management information by defining access point groups. The core groups that communicate can be in the same cell or in different cells.

Core group bridge overview

To configure communication between core groups, you must configure an *access point group*. An access point group is a collection of the core groups that communicate with each other. Add a *core group access point* to the access point group for each core group that needs to communicate.

A *core group access point* is a collection of server, node, and transport channel chain combinations that communicate for the core group. Each core group has one or more defined core group access points. The **DefaultCoreGroup** has one default core group access point. However, you might consider configuring more than one core group access point for a core group if that particular core group needs to be connected to other core groups that are on different networks. See “Advanced core group bridge configurations” on page 506 for more information about configuring core groups to communicate across different networks.

The node, server, and transport channel chain combinations that are in a core group access point are called *bridge interfaces*. A server that hosts the bridge interfaces is a *core group bridge server*. The transport channel chain defines the set of channels that are used to communicate with other core group bridge servers. Each transport channel chain has a configured port that the core group bridge server uses to listen for messages from other core group bridge servers.

Each core group access point must have at least one core group bridge server. The core group bridge server provides the bridge interface for each core group access point. Because core group bridge servers within a core group access point serve as backups for each other, it is recommended that you have two core group bridge servers within each core group access point. Then, if one core group bridge server fails, the other core group bridge server can take over the failed core group bridge server’s responsibilities.

If you are configuring communication between core groups that are in the same cell, create one access point group and add a core group access point for each core group that needs to communicate. See “Advanced core group bridge configurations” on page 506 for more information about configuring communication between core groups that are in the same cell.

If you are configuring the core group bridge between core groups that are in different cells, you still use an access point group. However, you must create and configure the access point group for each cell. Each cell has an access point group that contains a core group access point for the core group that is in the cell, and a *peer access point* for each peer cell.

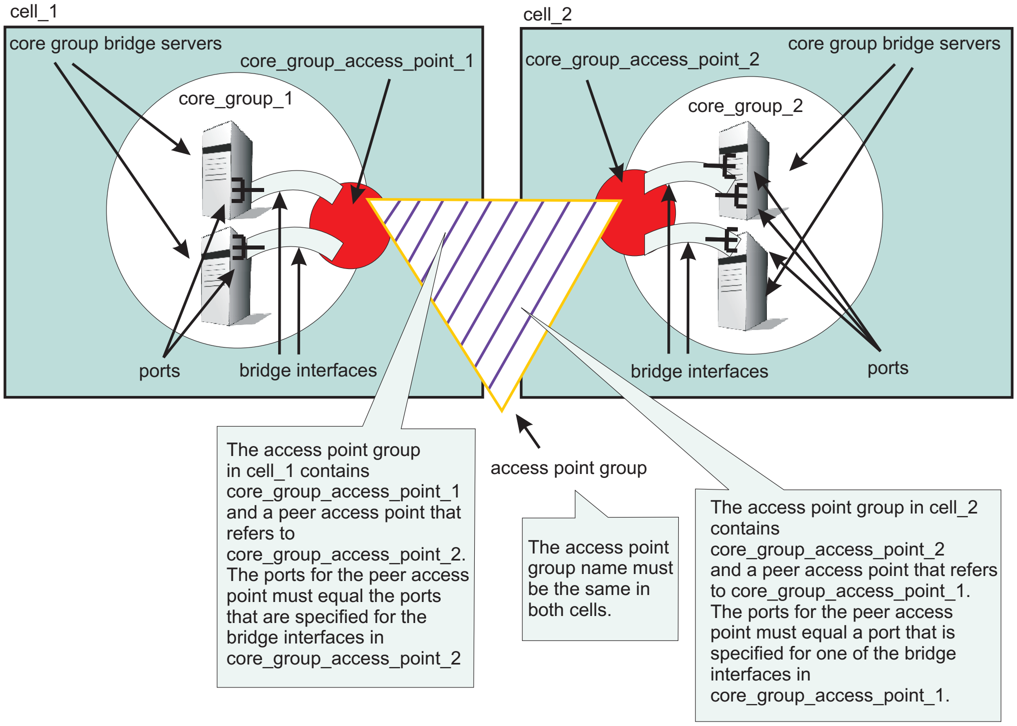
A peer access point references a core group access point that is configured in a different cell. Each access point group must have one peer access point for each different cell. Do not configure multiple peer access points that reference the same cell.

Each peer access point has one or more *peer ports* or one *proxy peer access point*.

A peer port corresponds to a bridge interface that is defined in the peer cell. You can define several peer ports for each peer access point.

Define a proxy peer access point if the peer access point cannot be reached directly by using a peer port, but can be reached by using another peer access point. The proxy peer access point specifies a peer access point that can communicate with the peer core group that cannot be reached directly. The proxy peer must have defined peer ports. Specify one proxy peer or one or more peer ports, but not both. See “Advanced core group bridge configurations” on page 506 for more information about proxy peer access points.

The following diagram shows a core group bridge configuration between two different cells that is using peer access points with peer ports.


cell_1
core group bridge servers
core_group_access_point_1
core_group_1
ports
bridge interfaces
cell_2
core_group_access_point_2
core group bridge servers
core_group_2
bridge interfaces
ports
access point group

The access point group in cell_1 contains core_group_access_point_1 and a peer access point that refers to core_group_access_point_2. The ports for the peer access point must equal the ports that are specified for the bridge interfaces in core_group_access_point_2

The access point group name must be the same in both cells.

The access point group in cell_2 contains core_group_access_point_2 and a peer access point that refers to core_group_access_point_1. The ports for the peer access point must equal a port that is specified for one of the bridge interfaces in core_group_access_point_1.

Figure 7. Core group bridge configuration in two different cells

Configuring the core group bridge between core groups that are in different cells

The core group bridge service can be configured for communication between core groups. Use an access point group to define the core groups that communicate. Use this task to configure communication between core groups that are in different cells.

Make sure that:

- You have two or more core groups that are in different cells. Core groups are a statically defined component of the high availability manager. See “Core groups” on page 441 for more information about core groups.
- Any cell that uses core group bridges to connect to core groups in other cells have a name that is unique when compared to the names of the other cells. See “Configuring communication between core groups that are in the same cell” on page 511 for more information.
- Any core group that uses a core group bridge is configured with Channel framework as its transport mechanism.

Use the core group bridge service to share the availability status of the servers in each core group among all the configured core groups. Enable the core group bridge only when the service is required by a WebSphere Application Server component.

When you configure core group communication between core groups that are in different cells, you must configure an access point group to connect the core groups. The name of the access point group must be the same in all of the connected cells. This task uses the DefaultAccessPointGroup access point group, but you can create and use another access point group.

You can define the CGB_ENABLE_602_FEATURES custom property on all of the access point groups in your configuration if you want to be able to add core group bridge servers to the configuration without restarting the other servers in the configuration. After you enable this property, you can add a core group bridge server in one cell, without modifying the configuration in the other cell to include peer ports for the core group bridge server. Instead of manually configuring peer ports in each of the cells, you can configure the peer ports in one cell and let the other cell discover the peer ports for the first cell.

If you decide to use the CGB_ENABLE_602_FEATURES custom property, you must decide which cell is the listener cell and which cell initiates contact with the other cells before you begin your configuration. The listener cell does not need to contain any peer access points or peer ports for the other cells in the configuration. For example, in a configuration that contains a secured cell and an unsecured cell, configure the unsecured cell as the listener. The unsecured cell cannot access information about the secured cell. Configure the core group bridge service on the listener cell first.

Attention: Do not configure the CGB_ENABLE_602_FEATURES custom property if you already configured the FW_PASSIVE_MEMBER custom property. Any server for which the configured the FW_PASSIVE_MEMBER custom property is configured cannot initiate contact with other systems in the configuration. For more information about the FW_PASSIVE_MEMBER custom property, see "Core group bridge custom properties" on page 517.

Complete the following set of steps for each of the cells in your configuration.

1. Configure bridge interfaces for your core group access point. Configuring a bridge interface indicates that the specified node, server, and chain combination is a core group bridge server. This node and server use the specified chain to communicate with other core groups.
   a. In the administrative console, click **Servers > Core groups > Core group bridge settings > Access point groups >** *DefaultAccessPointGroup* **> Core group access points >** *CGAP_1\DefaultCoreGroup* **> Show detail > Bridge interfaces > New**.
   b. Select a node, server, and transport channel chain that becomes your bridge interface.
   c. Click **Apply**.
   d. Repeat this set of steps to add more bridge interfaces to the core group access point. Define at least two bridge interfaces for each core group access point to back up your configuration. By defining two bridge interfaces, you define two core group bridge servers. If one core group bridge server fails, the other can take over the work so that the communication between the core groups can continue.

      Important:
      The bridge interfaces that you select must all have the same transport channel chain.
2. If you want the ability to add core group bridge servers to the configuration without restarting the other servers in the configuration, define the CGB_ENABLE_602_FEATURES custom property on all of the access point groups in your configuration.
   a. In the administrative console, click **Servers > Core groups > Core group bridge settings > Access point groups >** *DefaultAccessPointGroup* **> Custom properties > New**.
   b. Type the name as CGB_ENABLE_602_FEATURES and set the value to any string.

The existence of the CGB_ENABLE_602_FEATURES property enables the property. Therefore, you can set the value to any string value. Setting the value to false does not disable the property. To disable the property, you must remove it from the list of defined custom properties or change its name.

c. Click **Apply** and save your configuration.

When you complete this step on all the access point groups in your configuration, you can add a bridge interface to one of the cells. You can save the configuration so that it is propagated to all of the nodes. Instead of restarting all of the application servers, you need to restart the new bridge interface server only.

3. Add peer access points and peer ports to your access point group.

If you defined the CGB_ENABLE_602_FEATURES custom property for all of the access point groups in your configuration, you do not need to add peer access points or peer ports to the listener cell.

Add a peer access point for each core group that is in another cell. Within each peer access point, you should configure a peer port that corresponds to each bridge interface in the other cell. Before you add a peer access point, you should have the following information about the other cell:

- cell name
- core group name
- core group access point name
- host and port information. The host and port correspond to the bridge interfaces that are configured in the other cell. Specify a peer port for each bridge interface that is in the other cell.

a. In the administrative console, click **Servers > Core groups > Core group bridge settings > Access point groups >** *DefaultAccessPointGroup* **> Peer access points > New**.

b. Specify the information for your peer access point and click **Next**.

c. Select **Use peer ports**. Specify the host and port information for your peer cell. For example, if you defined a bridge interface in *cell_x*, use that configuration information for your peer port in *cell_y*.

d. Click **Next** and then **Finish**. Save your configuration.

If more than one bridge interface is defined in your peer cell, add additional peer ports for each bridge interface.

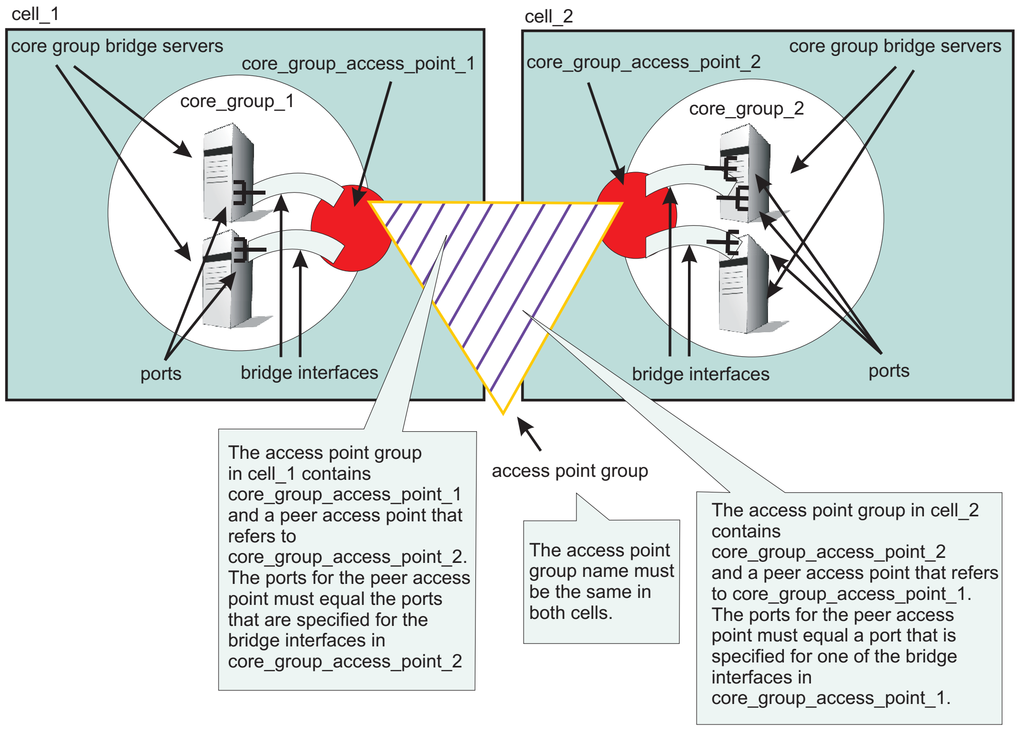
a. Click **Peer_access_point_name > Show detail > Peer ports > New**.

b. Enter the host name and port.

c. Click **Apply** and save your changes.

You configured the core group bridge between core groups that are in different cells.

The following illustration is an example of a configuration between two core groups that are in two different cells. Each cell has a defined *DefaultAccessPointGroup* access point group, which contains one core group access point for the core group that is in the cell and a peer access point for the other cell.


cell_1
core group bridge servers
core_group_access_point_1
core_group_1
ports
bridge interfaces
cell_2
core_group_access_point_2
core group bridge servers
core_group_2
bridge interfaces
ports
access point group

The access point group in cell_1 contains core_group_access_point_1 and a peer access point that refers to core_group_access_point_2. The ports for the peer access point must equal the ports that are specified for the bridge interfaces in core_group_access_point_2

The access point group name must be the same in both cells.

The access point group in cell_2 contains core_group_access_point_2 and a peer access point that refers to core_group_access_point_1. The ports for the peer access point must equal a port that is specified for one of the bridge interfaces in core_group_access_point_1.

Continue configuring the high availability environment. See Chapter 9, “Setting up a high availability environment,” on page 437 for more information.

Core group bridge settings

The core group bridge is the service that enables communication between core groups. A core group is a statically defined component of the high availability manager. Use this page to view the structure of your access point groups. Access point groups link core groups that are in the same cell or in different cells and allow the core groups to communicate with each other.

To view this administrative console page, click **Servers > Core groups > Core group bridge settings**.

Each access point group is a collection of core group access points. If you are configuring communication between core groups that are in the same cell, configure an access point group with a core group access point for each core group in the cell. If you are configuring communication between core groups in different cells, configure an access point group that has one core group access point for the local cell and a peer access point for each other cell.

Each core group access point has one or more bridge interfaces. Each peer access point has a proxy peer or one or more peer ports. A bridge interface is a server that is configured to communicate with other core groups by using a particular transport channel chain. Click **Access point groups** to configure the settings for each access point group that is configured.

- **Access point group** - An access point group defines the core groups that communicate with each other. Each access point group consists of a collection of core groups.
  - **Core group** - Specifies a core group that is in this access point group. Core groups are referenced by core group access points.
    - **Core group access point** - The core group access point defines the set of servers that provide access to the core group. Bridge interfaces define the servers that are in the core group access point.

- **Bridge interface** - A bridge interface defines a node, server, and chain for the core group access point. The chain defines the transport channels that the server uses for receiving information. Typically, all the bridge interfaces in a core group access point use the same chain.

  Important: The bridge interfaces listed in the display tree on this page are listed as `*,port`. The asterisk symbol represents the multi-home support for DCS instead of a specific hostname.

– **Peer core group** - Specifies a core group in a different cell. Define peer access points to communicate with peer core groups.

  - **Peer access point** - Each peer access point is used to communicate a core group in a different cell. Each peer access point corresponds to a core group access point that is in the peer cell. Use one or more peer ports or one proxy peer to define the communication settings.
    - **Peer port** - Each peer port identifies a bridge interface of a core group bridge in the peer cell.
    - **Proxy peer** - A proxy peer is used to identify the communication settings for a peer access point that cannot be accessed directly through peer ports. A proxy peer specifies a peer access point that can communicate with the destination core group. The specified proxy peer must be a peer access point that has defined ports.

Access point group collection

Use this page to view the sets of access point groups. Access point groups define the set of core groups that communicate with each other. Access point groups that connect multiple cells must have one core group access point and a single peer access point for each remote cell. Access point groups that provide communications between core groups in the same cell must contain only core group access points.

To view this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups**.

Name:

Specifies the name of the access point group. The access point group name must be unique within the cell.

Access point group settings:

Use this page to modify the core group access points and the peer access points that belong to this access point group. An access point group defines the set of core groups that communicate with each other. Group the access points to support communication. Access points can be either peer access points or core group access points. Define core group access points so that core groups in the same cell can communicate. Define peer access points so that core groups in different cells can communicate.

To view this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups >** *access_point_group_name*.

From this page, you can edit the core group access points or the peer access points that belong to the access point group.

Name:

Specifies the name of the access point group. The access point name must be unique within a cell.

Core group access point collection

Use this page to configure your set of core group access points. Core group access points define the set of servers that provide access to the core group. At least one core group access point must be defined for each core group in the local cell.

To access this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups >** *access_point_group_name* **> Core group access points**.

Available core group access points:

A list of core group access points that are available to add to the access point group.

Core group access points in *access point group**:***

A list of core group access points that are in the specified *access point group*.

Core group access point settings:

Use this page to configure your core group access points. The core group access point defines the set of core group bridge servers that provide access to the core group. A core group bridge server is an application server that is configured to run the core group bridge service. Define unique core group access points for each different network over which you want the core group in this cell to connect to other core groups that are defined in another cells. The core group access point has a collection of bridge interfaces. Each server that is used as a bridge must have a unique bridge interface for every core group access point in the core group.

For example if you define access points AccessPointA and AccessPointB and ServerX and ServerY are configured as core group bridges in a core group, ServerX must have unique bridge interfaces for both AccessPointA and AccessPointB. The ServerY server must also have unique bridge interfaces for both AccessPointA and AccessPointB.

When a you create a core group, a core group access point is automatically created. Do not delete the last access point in a core group.

The core group access point that is automatically defined belongs to a default access point group. You can use the default core group access point and access point group to configure communication between core groups. You must create and configure bridge interfaces for the default core group access point. See "Bridge interface settings" on page 503 for more information about bridge interfaces.

To access this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups >** *access_point_group_name* **> Core group access points** **>***core_group_access_point_name* > **Show detail**.

Name:

Specifies the name for the core group access point.

Core group:

Specifies the core group that is associated with this core group access point.

Bridge interface collection

Each core group access point has a collection of bridge interfaces. This collection defines the interfaces on the set of servers that provide access to the core group. All the servers in this collection have the core group bridge service enabled. The core group bridge service provides communication between core groups. A bridge interface defines the node, the server, and the chain for the core group access point. The chain defines the transport channels that are used by the server for receiving information.

To access this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups >** *access_point_group_name* **> Core group access points >***access_point_name* **> Show detail > Bridge interfaces**.

Server:

Specifies the node and the server combinations that are bridge interfaces for the core group access point.

Transport channel chain:

Specifies the transport channel chain that is used for transport by the bridge interface. For all the core group access points in an access point group, the transport channel chains must resolve to the same host. To ensure that the transport channel chains resolve to the same host, use the same chain for all of the core group access points in an access point group.

Bridge interface settings:

Use this page to specify the bridge interfaces that provide access to the core group access point. A bridge interface is a particular node and server that runs the core group bridge service. The core group bridge service is the service that provides communication between core groups.

A bridge interface is defined by a unique combination of a node, server, and transport chain. You cannot configure a cluster of servers to run the same bridge interface. A transport chain represents a network protocol stack that is operating within an application server.

To access this administrative console page, click **Servers** > **Core groups** > **Core group bridge settings** > **Access point groups** > *access_point_group_name* > **Core group access points** > *access_point_access_point_name* > **Show detail** > **Bridge interfaces** > *server_node*.

Bridge interfaces: Select one of the listed server, node, and chain combinations that are available to become bridge interfaces for your core group access point.

Bridge interface creation:

A bridge interface specifies a particular node and server that runs the core group bridge service. A bridge interface is defined by a unique combination of a node, a server, and a transport chain. A transport chain represents a network protocol stack that is operating within an application server.

To access this administrative console page, click **Servers** > **Core groups** > **Core group bridge settings** > **Access point groups** > *access_point_group_name* > **Core group access points** >*access_point_name* > **Show detail** > **Bridge interfaces** > **New**.

Available bridge interfaces:

Specifies the node, server and transport channel chain combinations that are available to become bridge interfaces for this core group access point. Only bridge interfaces with transport chains that contain Transmission Control Protocol (TCP) inbound channels using the same port name as existing bridge interfaces in this core group access point are displayed. The bridge interfaces that are already used by any core group access point are not displayed.

Peer access point collection

Use this page to view a list of peer access points. Peer access points are used to communicate with core groups that are in other cells. A peer access point collection is the set of peer access points that are used to communicate with the core group access point in this access point group. Specify a single peer access point for each remote cell. This collection cannot contain two peer access points for the same cell.

To view this administrative console page, click **Servers** > **Core groups** > **Core group bridge settings** > **Access point groups** > *access_point_group_name* > **Peer access points**.

Available peer access points:

Specifies a list of peer access points that are available to join the access point group.

***Peer access points in** access_point_group:* Specifies a list of peer access points that are in the selected access point group.

Peer access point settings:

Use this page to configure a peer access point. Each peer access point is used to communicate with core groups in other cells. A peer access point corresponds to a core group access point in the peer cell. The peer access point communication settings are specified by using one or more peer end points or a proxy peer.

A peer access point must contain either peer ports or a proxy peer access point, but not both. When the peer access point is directly accessible within its access point group, specify peer ports. When the peer access point can be reached only indirectly, use a proxy peer access point. A proxy peer access point is used to identify the communication settings for the peer access point that cannot be accessed directly. The proxy peer access point specifies a peer access point that can communicate with the appropriate destination core group. The specified proxy peer access point must be a peer access point that has defined ports.

To view this administrative console page, click **Servers** > **Core groups** > **Core group bridge settings** > **Access point groups** > *access_point_group_name* > **Peer access points** > *peer_access_point_name* > **Show detail**.

Name:

Specifies the name of the peer access point. The name must be unique within the local cell.

Cell:

Specifies the cell in which the peer access point resides.

Core group:

Specifies the core group in which the peer access point resides.

Core group access point:

Specifies the name of the core group access point that is in the peer cell.

default	defaultCoreGroupAccessPoint

Use peer ports:

Specifies that you are using peer ports instead of a proxy peer access point. Use peer ports when the peer access point is directly accessible within its access point group. Click **Peer ports** to specify the peer ports for the peer access point.

Use proxy peer access point:

Specifies that you are using a proxy peer access point instead of peer ports. A proxy peer is defined when the peer access point can be reached only indirectly through another peer access point. A proxy peer is used to identify the communication settings for the peer access point that cannot be accessed directly. The proxy peer specifies a peer access point that can communicate with the destination core group. The specified proxy peer must be a peer access point that has defined peer ports.

Proxy peer access point:

Specifies the specific peer access point that is used to access a core group.

Peer port collection

Use this page to define the peer ports for the peer access point. Each peer port identifies a bridge interface of a core group bridge service in the peer cell. Each peer access point that does not have a proxy peer must have one or more peer ports.

To view this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups >** *access_point_group_name* **> Peer access points >** *peer_access_point_name* **> Show detail > Peer ports**.

Host:

Specifies the host name that is used by the bridge interface in the remote cell.

Port:

Specifies the port that is used by the bridge interface in the remote cell.

Peer port settings:

Use this page to configure a peer port. A peer port identifies the host name and port of an application server that is a bridge interface in another cell. This application server is using the core group bridge service to communicate with other core groups. Each peer access point can have one or more peer ports. Each port identifies a bridge interface of a core group bridge service in the peer cell.

To view this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups >** *access_point_group_name* **> Peer access points >** *peer_access_point_name* **> Show detail > Peer ports >** *peer_port_name*.

Host:

Specifies the host name on which the core group bridge in the remote cell is listening.

Port:

Specifies the port number that is associated with the host on which the core group bridge in the remote cell is listening.

Creating advanced core group bridge configurations

Use this task to configure core groups to communicate with each other.

Configure two or more core groups that need to communicate with each other. For more information about core groups, see “Core groups” on page 441. To configure core groups, see “Creating a new core group” on page 478.

If you are configuring core group communication between core groups that are in different cells, see “Configuring the core group bridge service” on page 495. Communication between core groups that are in different cells is the most common usage scenario. Using this task, you can configure core group communication between core groups that are in the same cell or that use a proxy peer to communicate across 3 cells.

- Configure core group communication between core groups that are in the same cell. Configuring communication between any core groups that are in the same cell is required. For more information, see “Configuring communication between core groups that are in the same cell” on page 511.

- Configure communication between core groups using a proxy peer access point. Sometimes, your core group might not have access to the core group that you want to communicate with. However, if you can access a core group that can communicate with the inaccessible core group, you can create a proxy peer access point. For more information, see “Configuring core group communication using a proxy peer access point” on page 514.

Multiple core groups can communicate with each other.

Continue configuring the high availability environment. See Chapter 9, “Setting up a high availability environment,” on page 437 for more information.

Advanced core group bridge configurations

This topic describes advanced core group bridge configurations. These configurations are not performed as often as the typical core group bridge between core groups that are in different cells.

Advanced configuration scenarios

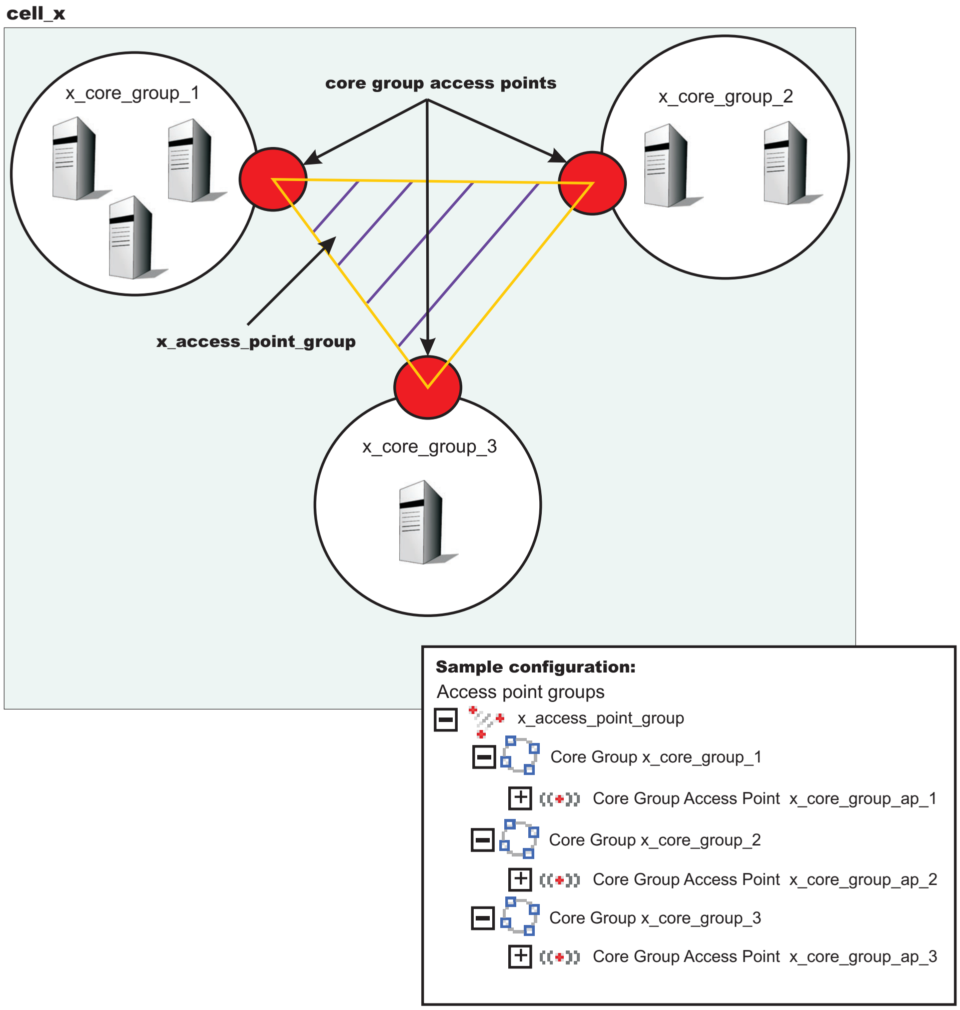
The most common core group bridge configuration is between two core groups that are in different cells on a single network. See “Core group communications using the core group bridge service” on page 495 for more information about this common scenario. The scenarios that are described in this topic are for advanced configuration situations.

There are four types of communication between core groups that you can configure:

- Communication between core groups that are in the same cell
- Communication within the cell and outside of the cell
- Communication between core groups across different networks
- Communication between core groups using a proxy peer access point

Communication between core groups that are in the same cell

All core groups that are in the same cell must be configured to communicate with each other. To configure core group communication within a cell, create one access point group with one core group access point for each core group. Select one or more servers to be core group bridge servers, and define a bridge interface for each server. All the bridge interfaces that are in an access point group that connects core groups that are in the same cell must have a node, server, and chain combination that resolves to the same port. To make sure all the bridge interfaces resolve to the same port, you can configure all the bridge interfaces use the same chain name. The following image shows an example of three core groups that are in the same cell and are connected by one access point group. The sample configuration shows how communication between core groups in the same cell is configured in the administrative console.



Sample configuration:

Access point groups

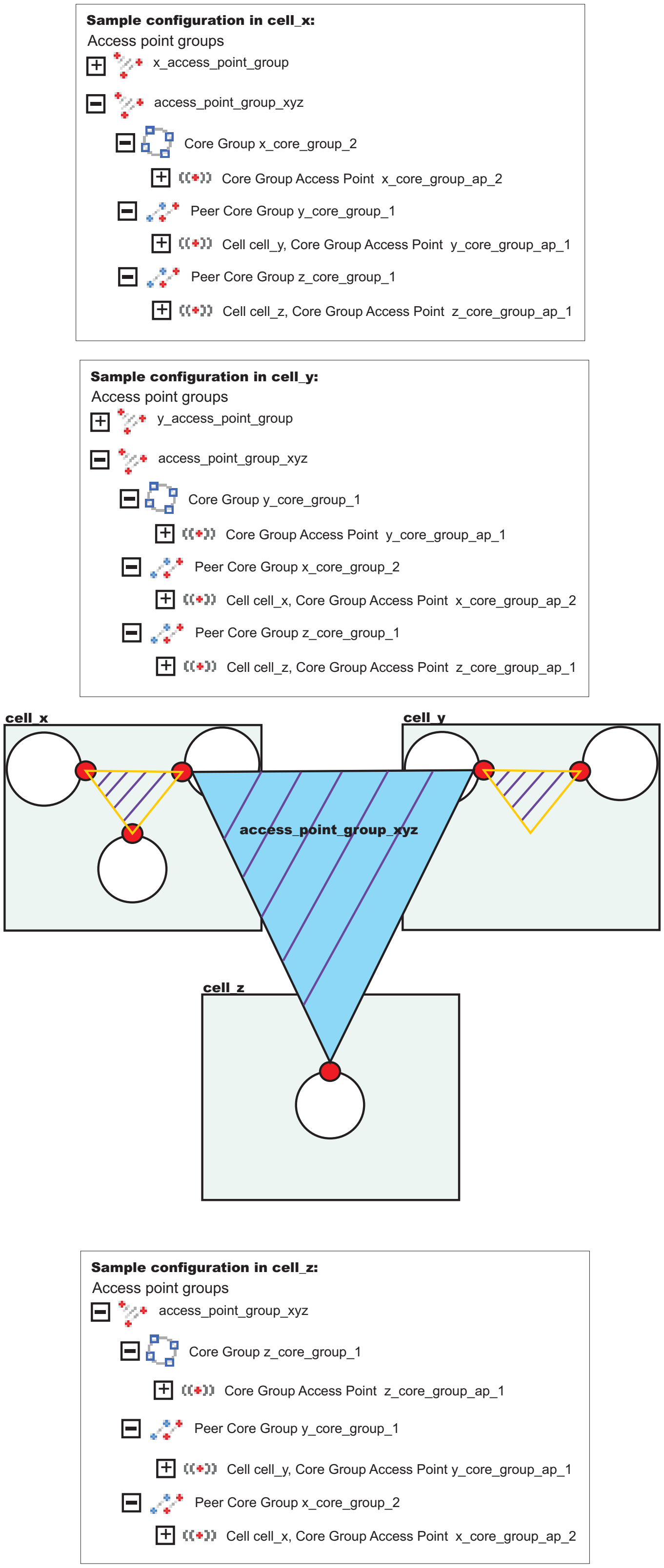
- x_access_point_group
  - Core Group x_core_group_1
    - Core Group Access Point x_core_group_ap_1
  - Core Group x_core_group_2
    - Core Group Access Point x_core_group_ap_2
  - Core Group x_core_group_3
    - Core Group Access Point x_core_group_ap_3

Figure 8. Communication between core groups that are in the same cell

See “Configuring communication between core groups that are in the same cell” on page 511 for more information.

Communication within the cell and outside of the cell

The following example illustrates a configuration between three core groups that are in three different cells. Each cell has one access point group for communication between core groups in the cell. Each cell also has a defined *access_point_group_xyz* access point group, which contains one core group access point group for the core group that is in the cell, and one core group access point for each of the core groups in the other two cells.



Sample configuration in cell_x:

Access point groups

⊞ x_access_point_group

⊟ access_point_group_xyz

 ⊟ Core Group x_core_group_2

  ⊞ Core Group Access Point x_core_group_ap_2

 ⊟ Peer Core Group y_core_group_1

  ⊞ Cell cell_y, Core Group Access Point y_core_group_ap_1

 ⊟ Peer Core Group z_core_group_1

  ⊞ Cell cell_z, Core Group Access Point z_core_group_ap_1

Sample configuration in cell_y:

Access point groups

⊞ y_access_point_group

⊟ access_point_group_xyz

 ⊟ Core Group y_core_group_1

  ⊞ Core Group Access Point y_core_group_ap_1

 ⊟ Peer Core Group x_core_group_2

  ⊞ Cell cell_x, Core Group Access Point x_core_group_ap_2

 ⊟ Peer Core Group z_core_group_1

  ⊞ Cell cell_z, Core Group Access Point z_core_group_ap_1

Sample configuration in cell_z:

Access point groups

⊟ access_point_group_xyz

 ⊟ Core Group z_core_group_1

  ⊞ Core Group Access Point z_core_group_ap_1

 ⊟ Peer Core Group y_core_group_1

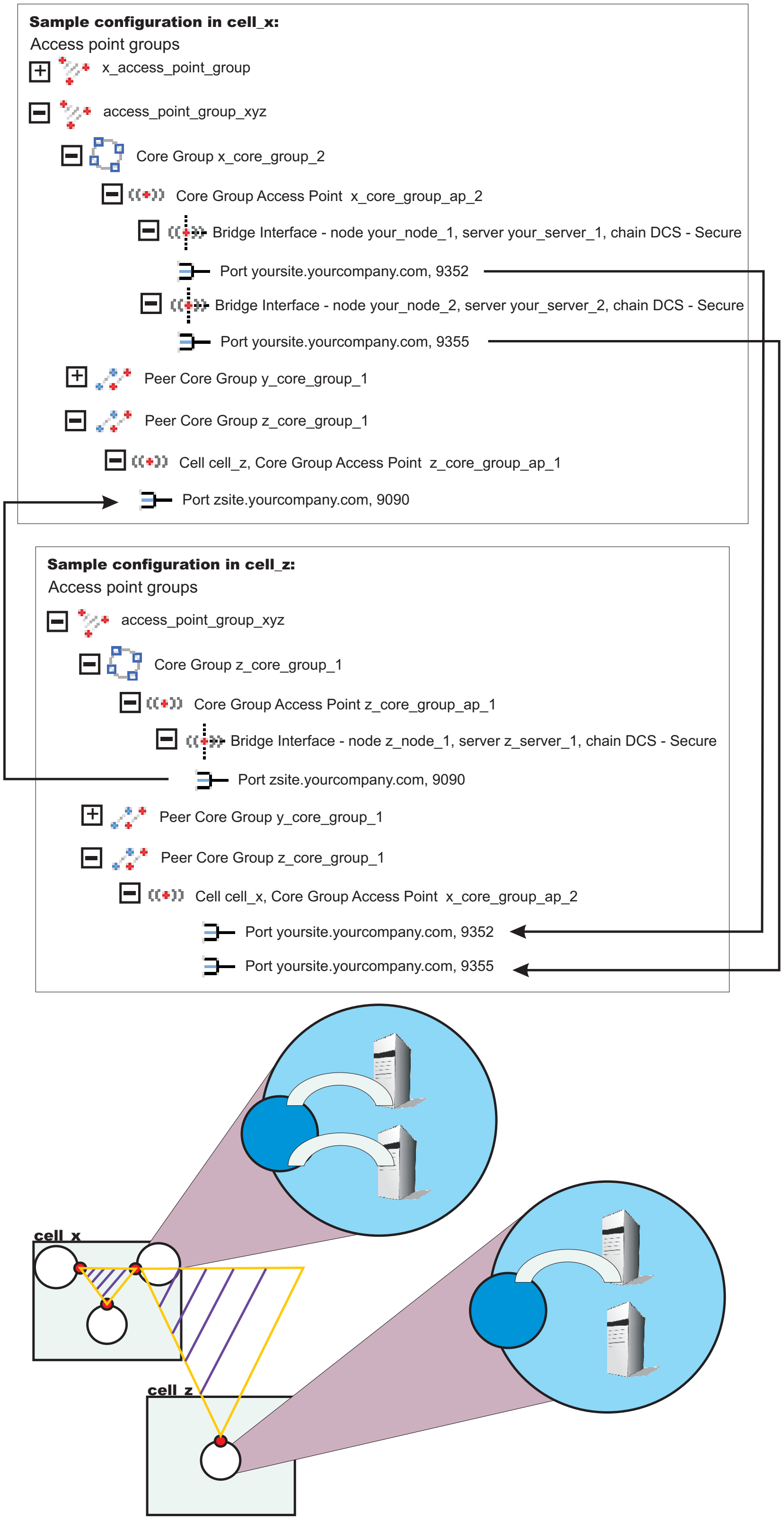
  ⊞ Cell cell_y, Core Group Access Point y_core_group_ap_1

 ⊟ Peer Core Group x_core_group_2

  ⊞ Cell cell_x, Core Group Access Point x_core_group_ap_2

Figure 9. Communication between core groups that are in the same cell with core groups outside of the cell

The following example shows the relationship between bridge interfaces and peer ports for the communication between the*cell_x* cell and the *cell_z* cell. In the *cell_x* cell, two bridge interfaces are defined. In the *cell_z* cell a peer access point exists for the *x_core_group_ap_2* core group access point with peer ports defined that correspond to the bridge interface information that is defined in the *cell_x* cell .



Sample configuration in cell_x:

Access point groups

- x_access_point_group
- access_point_group_xyz
  - Core Group x_core_group_2
    - Core Group Access Point x_core_group_ap_2
      - Bridge Interface - node your_node_1, server your_server_1, chain DCS - Secure
        - Port yoursite.yourcompany.com, 9352
      - Bridge Interface - node your_node_2, server your_server_2, chain DCS - Secure
        - Port yoursite.yourcompany.com, 9355
  - Peer Core Group y_core_group_1
  - Peer Core Group z_core_group_1
    - Cell cell_z, Core Group Access Point z_core_group_ap_1
      - Port zsite.yourcompany.com, 9090

Sample configuration in cell_z:

Access point groups

- access_point_group_xyz
  - Core Group z_core_group_1
    - Core Group Access Point z_core_group_ap_1
      - Bridge Interface - node z_node_1, server z_server_1, chain DCS - Secure
        - Port zsite.yourcompany.com, 9090
  - Peer Core Group y_core_group_1
  - Peer Core Group z_core_group_1
    - Cell cell_x, Core Group Access Point x_core_group_ap_2
      - Port yoursite.yourcompany.com, 9352
      - Port yoursite.yourcompany.com, 9355

Figure 10. Bridge interfaces in one cell correspond to peer ports in the other cell

As a result, the *core_group_x* , *core_group_y* and *core_group_z* core groups can communicate with each other.

Communication between core groups across different networks

In this scenario, one core group is configured to communicate with two or more core groups in different cells across two or more networks. For example, a core group in the *cell_x* cell needs to communicate with core groups in the *cell_y* and *cell_z* cells. Create two access point groups in the *cell_x* cell. The*access_point_group_xy* access point group, in the *cell_x* cell contains a core group access point and a peer access point for the core group in the *cell_y* cell. The *access_point_group_xz* access point group in the *cell_x* cell contains a core group access point and a peer access point for the core group in the *cell_z* cell. The *cell_y* cell has an *access_point_group_xy* access point group, which has a core group access point and a peer access point for the *cell_x* cell. The *cell_z* cell has an *access_point_group_xz*access point group, which has a core group access point and a peer access point for the *cell_x* cell.

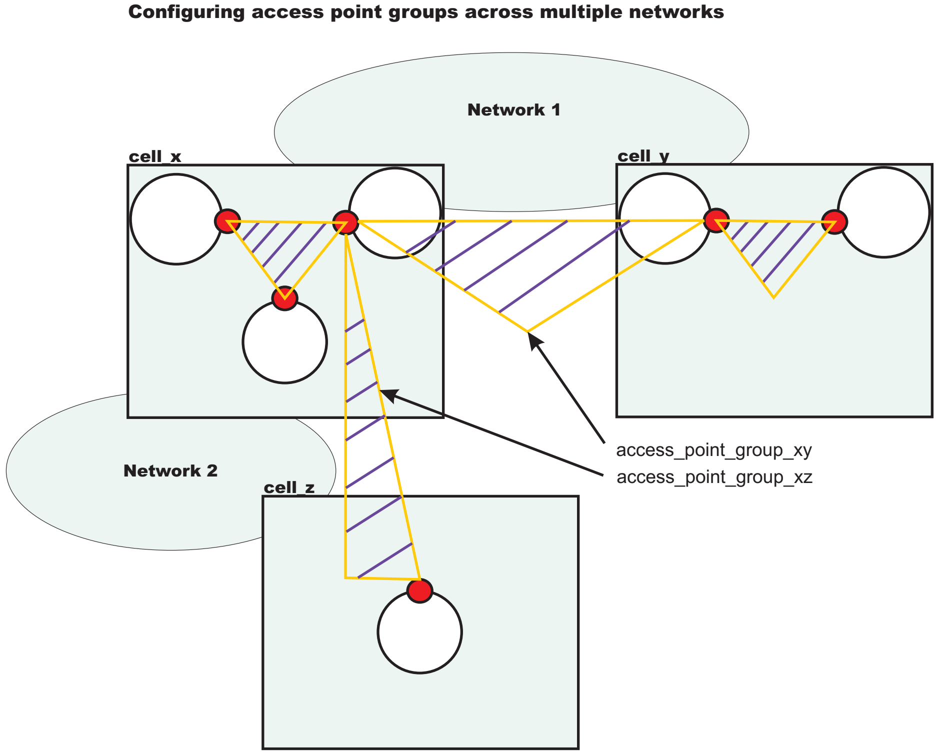


Figure 11. Core group communication across different networks

Communication between core groups using a proxy peer access point

Use a proxy peer when the core groups cannot directly communicate. The two core groups must have access to a single core group that can pass information between the two core groups. To understand what a proxy peer access point does, consider a connecting flight when flying on an airplane. To fly from Pittsburgh to London you first have to fly to New York City, where you change planes and then fly to London. New York City is the *proxy peer access point* for London.

When defining a proxy peer, the *x_core_group_2* core group in the *cell_x* cell cannot communicate directly with the core group in the *cell_z* cell. However, both core groups can communicate with the core group in the *cell_y* cell. To configure communication between the *cell_x* cell and the*cell_z* cell, you must configure two access point groups. The core group access point in the *cell_y* cell is in both the *access_point_group_xy* and *access_point_group_yz* access point groups. The following image shows an overview of a proxy peer configuration.

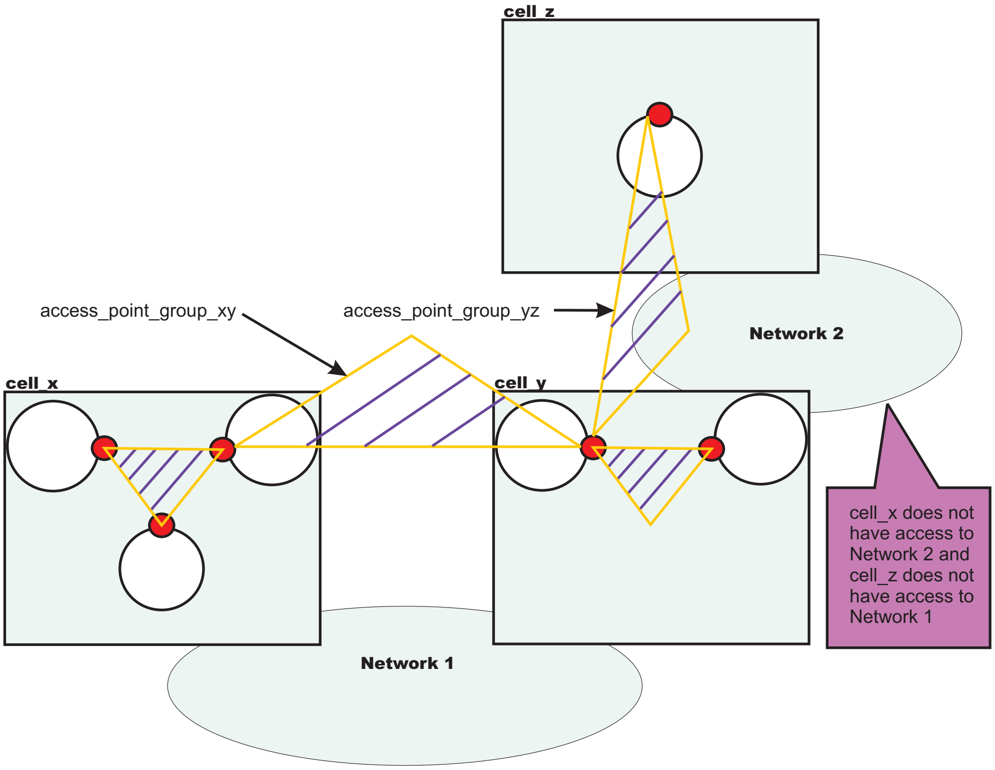

cell_z
access_point_group_xy
access_point_group_yz
Network 2
cell_x
cell_y
cell_x does not have access to Network 2 and cell_z does not have access to Network 1
Network 1

Figure 12. Core group communication using a proxy peer access point

See “Configuring core group communication using a proxy peer access point” on page 514 for more information.

Configuring communication between core groups that are in the same cell

Use this task to configure communication between core groups that are in the same cell.

Configure two core groups with application servers that are in the same cell. For more information about when and how to configure multiple core groups, see “Creating a new core group” on page 478.

You must configure the core group bridge when you have multiple core groups that are in the same cell. Use the core group bridge service to share the availability status of the servers in each core group among all the configured core groups.

To configure communication between core groups, define an access point group. An access point group defines the core groups that can communicate with each other. Each access point group has one core group access point for each core group. Select one or more servers to be core group bridges, and define a bridge interface for each server.

See “Advanced core group bridge configurations” on page 506 for more information about the communication between core groups that are in the same cell.

1. Configure an access point group to define the core groups that need to communicate. An access point group contains the core group access points for the core groups that need to communicate. Core group access points define the set of servers that provide access to the core group. To configure communication between core groups that are in the same cell, you can choose an existing access point group or create a new access point group. To create an access point, complete the following steps:

a. In the administrative console, click **Servers > Core groups > Core group bridge settings > Access point groups > New**.

b. Enter a name for the access point group that is unique within the cell.

c. Add core group access points to your access point group. Choose any available core group access points for the core groups that need to communicate in the cell. A default core group access point is created whenever a core group is created. The access point group that you create must have a core group access point for each core group in the cell.

Restriction: Do not add any peer access points. Add peer access points only if you are configuring communication with a core group in a different cell. If you need to communicate with core groups that are outside of the cell, you must create another access point group that has one core group access point and one or more peer access points. See "Creating advanced core group bridge configurations" on page 505 for more information.

If you use an existing access point group, choose an access point group that does not have peer access points. To configure an existing access point group, perform the following steps:

a. In the administrative console, click **Servers > Core groups > Core group bridge settings**. Your current configuration with any existing access point groups is displayed.

b. Verify that the access point group does not have any peer access points. Peer access point groups are used for communication between core groups in different cells. Click the access point group you want to configure and ensure that no peer access points are listed.

c. Click **Access point groups >** *access_point_group_name* **> Core group access points**.

d. Add core group access points to your access point group. Choose any available core group access points for the core groups that need to communicate. The access point group you create should have a core group access point for each core group in the cell.

2. Create bridge interfaces for each core group access point. The bridge interfaces that you add provide access to the core group. Create at least one bridge interface for each core group access point. To provide high availability for the core group access point, configure two or more bridge interfaces. If a core group has multiple core group access points, each core group access point must contain the same number of bridge interfaces for the same set of servers. To configure bridge interfaces, perform the following steps:

   a. In the administrative console, click **Servers > Core groups > Core group bridge settings > Access point groups >** *access_point_group_name* **> Core group access points**.

   b. Click a core group access point in the access point group. Click **Show Detail**.

   c. To create a new bridge interface, click **Bridge interfaces > New**.

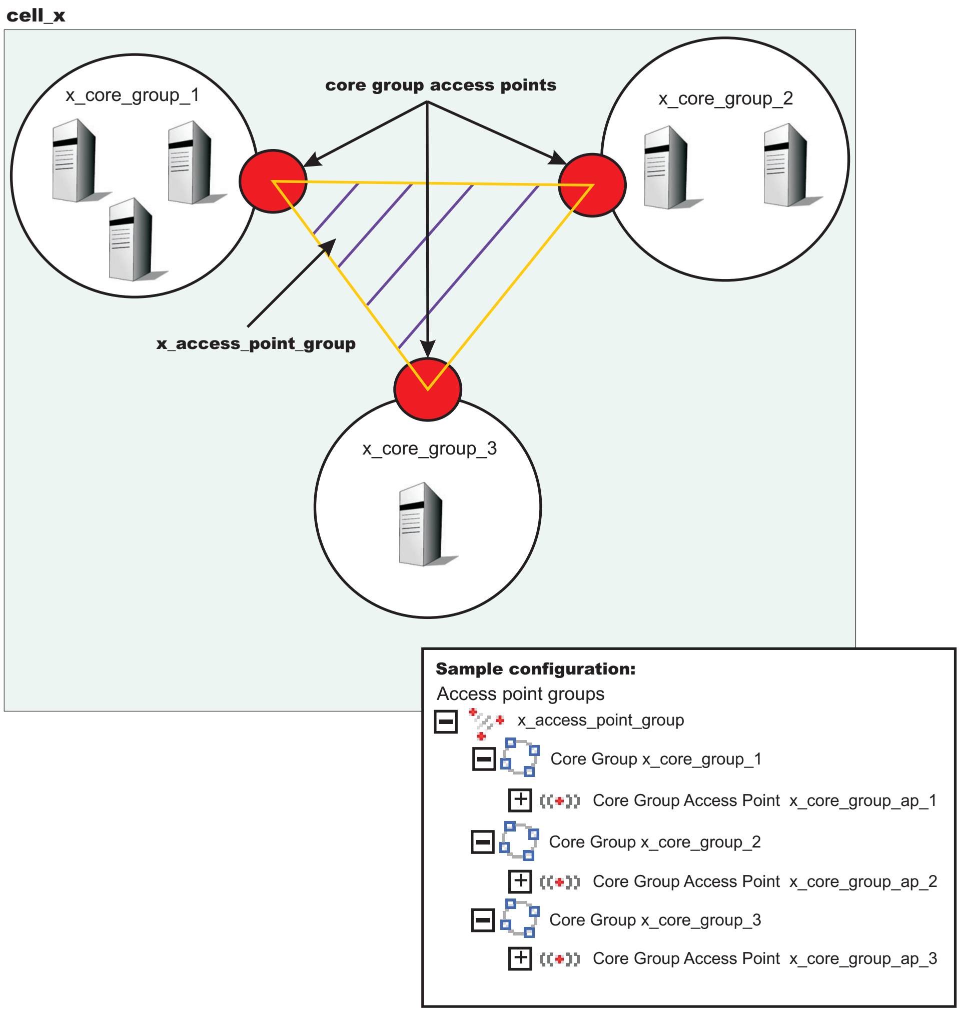
   d. Select a node, server, and transport channel chain combination for the bridge interface. Click **OK**. All the bridge interfaces for the core group access points that are in the same access point group must have transport channel chains with the same port name. You can configure the same port name by selecting the same chain name for all of the bridge interfaces. The transport channel chain can be the DCS or DCS-secure channel chains that are created for the DCS_UNICAST_ADDRESS transport chain.

   e. Consider creating at least two bridge interfaces for each access point. If one bridge interface fails, the other can still be active.

   f. Repeat these steps to create bridge interfaces for each core group access point in your access point group.

The core groups that are in the same cell and configured in an access point group can communicate.

In the *cell_x* cell, there are the *x_core_group_1*, *x_core_group_2*, and *x_core_group_3* core groups. Each core group already has a core group access point. The following image illustrates an access point group between the core groups in the *cell_x* cell and an example of the configuration in the administrative console.



Sample configuration:

Access point groups

⊟ x_access_point_group

⊟ Core Group x_core_group_1

⊞ Core Group Access Point x_core_group_ap_1

⊟ Core Group x_core_group_2

⊞ Core Group Access Point x_core_group_ap_2

⊟ Core Group x_core_group_3

⊞ Core Group Access Point x_core_group_ap_3

Figure 13. Three core group access points in the same cell belong to the same access point group.

Perform the following steps to configure communication between the three core groups in the *cell_x* cell:

1. Create the x_access_point_group access point group. Add a core group access point to the access point group for each core group that is in the cell. In this example, add the x_core_group_ap_1 , x_core_group_ap_2, and x_core_group_ap_3 access points to the x_access_point_group access point group.
2. Create bridge interfaces for each core group access point. The following diagram illustrates the bridge interfaces for thex_core_group_ap_2 core group access point:

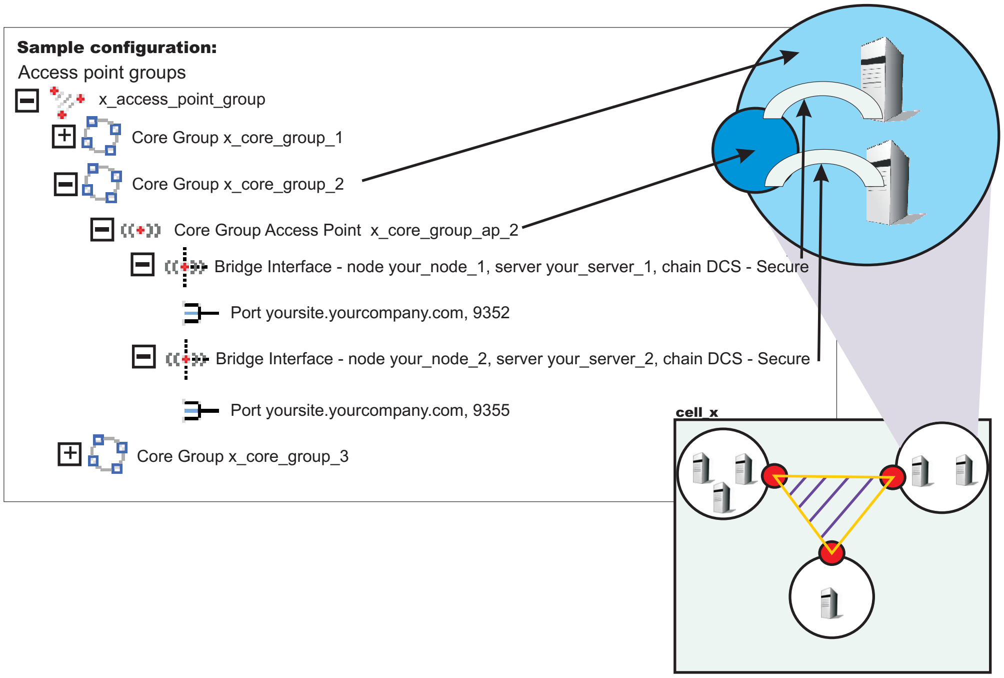

Sample configuration:
Access point groups
x_access_point_group
Core Group x_core_group_1
Core Group x_core_group_2
Core Group Access Point x_core_group_ap_2
Bridge Interface - node your_node_1, server your_server_1, chain DCS - Secure
Port yoursite.yourcompany.com, 9352
Bridge Interface - node your_node_2, server your_server_2, chain DCS - Secure
Port yoursite.yourcompany.com, 9355
Core Group x_core_group_3
cell_x

Figure 14. Core group access points contain one or more bridge interfaces.

Create two or more bridge interfaces for each core group access point.

By creating an access point group and adding all core groups in the cell to the access point group, you enabled communication between all the core groups that are in the *cell_x* cell.

You can configure this cell to communicate with core groups in other cells. See “Configuring the core group bridge between core groups that are in different cells” on page 497 and “Configuring core group communication using a proxy peer access point” for more information.

Configuring core group communication using a proxy peer access point

Use this task to configure communication between two core groups when they cannot communicate with each other directly through peer ports.

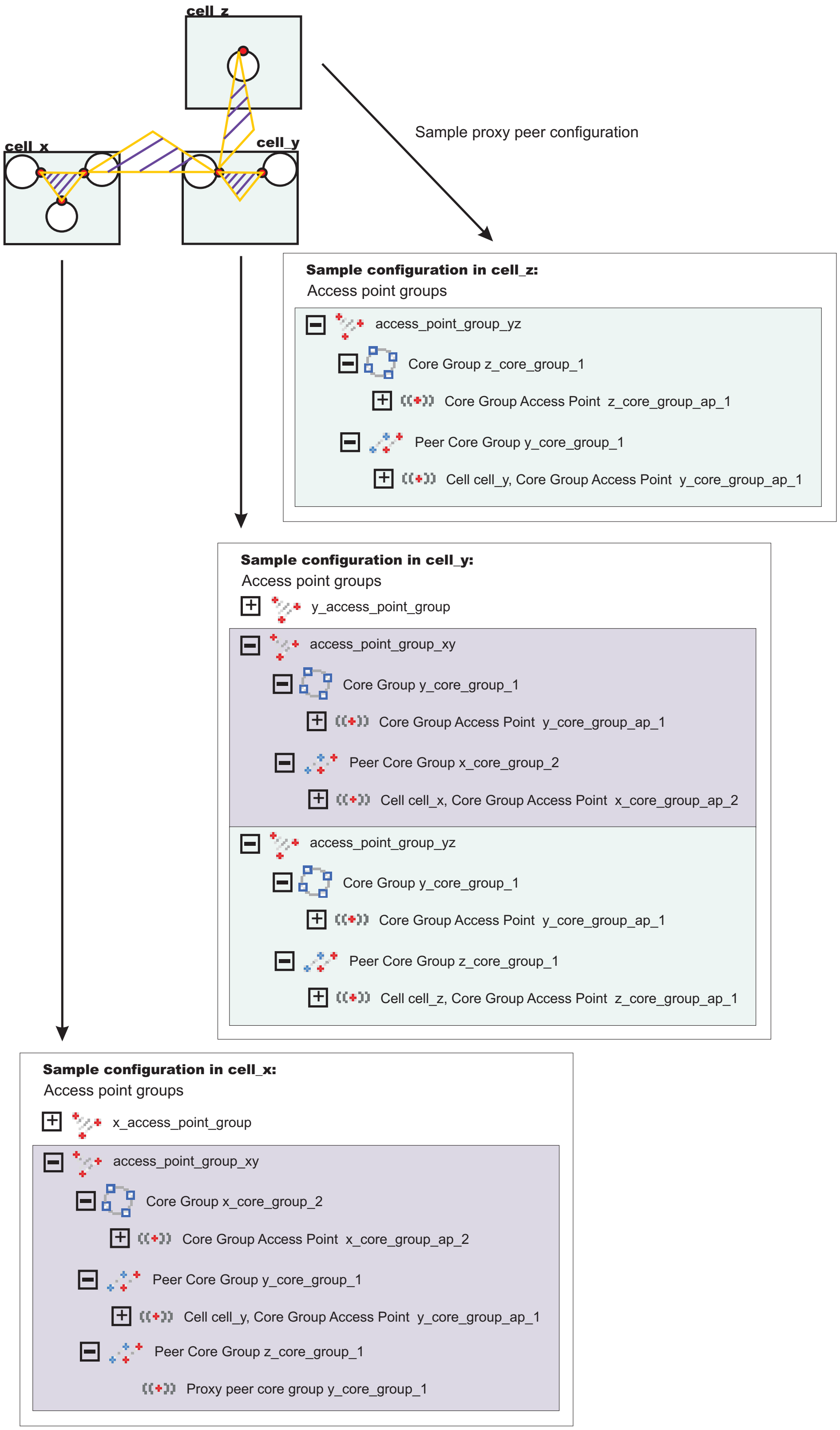
Configure a proxy peer access point to communicate with a core group if you cannot use peer ports. Use a core group that has configured a peer access point with peer ports to the core group that you want to communicate with. Before completing this task, make sure that you have access to a core group that can communicate with the core group that you cannot communicate with directly. To configure communication between core groups that are in different cells, see “Configuring the core group bridge between core groups that are in different cells” on page 497. If there are multiple core groups in each of the cells, they must be configured to communicate with each other. See “Configuring communication between core groups that are in the same cell” on page 511 for more information.

Use a proxy peer to communicate with a core group that you cannot access directly. This task describes how to configure a proxy peer with three core groups that are each in different cells. The *core_group_x* and *core_group_y* core groups can communicate directly with each other through peer ports. The *core_group_y* and *core_group_z* core groups can also communicate with each other though peer ports. However, the *core_group_x* core group cannot communicate with the *core_group_z* core group. To establish that communication, the *core_group_x* core group has a peer access point that is a proxy peer. The proxy peer is the peer access point to the *core_group_y* core group. For more information, see “Advanced core group bridge configurations” on page 506.

1. Configure the *core_group_x* and *core_group_y* core groups to communicate with each other by creating an access point group. For more information, see “Configuring the core group bridge between core groups that are in different cells” on page 497.
2. Configure the *core_group_y* and *core_group_z* core groups to communicate with each other by creating another access point group. For more information, see “Configuring the core group bridge between core groups that are in different cells” on page 497. When you do this step, create a second access point group in cell 2. The *core_group_2* core group communicates with both the *cell_x* and *cell_z* cells over two different networks.
3. Configure a peer access point that has a proxy peer. Create a new peer access point in the access point group that you created between the *core_group_x* and *core_group_y* core groups.
   a. In the administrative console, click **Servers > Core groups > Core group bridge settings > Access point groups >** *access_point_group_name* **> Peer access points**. Create a new peer access point, or select an existing access point.
   b. Enter a unique name for the peer access point. For the other cell, core group, and core group access point values, use the properties of the*core_group_z* core group.
   c. Click **Use a proxy peer access point**. Select the proxy peer access point that is the peer access point that you created in the *cell_x* cell that refers to the core group access point in the *cell_y* cell.

The *core_group_x* core group can communicate with the *core_group_z* core group by using a proxy peer.

The following example shows the configurations in each cell when you configure communication between the *cell_x* and *cell_z* cells using a proxy peer access point:



Sample configuration in cell_z:
Access point groups

- ⊟ access_point_group_yz
  - ⊟ Core Group z_core_group_1
    - ⊞ Core Group Access Point z_core_group_ap_1
  - ⊟ Peer Core Group y_core_group_1
    - ⊞ Cell cell_y, Core Group Access Point y_core_group_ap_1

Sample configuration in cell_y:
Access point groups

- ⊞ y_access_point_group
- ⊟ access_point_group_xy
  - ⊟ Core Group y_core_group_1
    - ⊞ Core Group Access Point y_core_group_ap_1
  - ⊟ Peer Core Group x_core_group_2
    - ⊞ Cell cell_x, Core Group Access Point x_core_group_ap_2
- ⊟ access_point_group_yz
  - ⊟ Core Group y_core_group_1
    - ⊞ Core Group Access Point y_core_group_ap_1
  - ⊟ Peer Core Group z_core_group_1
    - ⊞ Cell cell_z, Core Group Access Point z_core_group_ap_1

Sample configuration in cell_x:
Access point groups

- ⊞ x_access_point_group
- ⊟ access_point_group_xy
  - ⊟ Core Group x_core_group_2
    - ⊞ Core Group Access Point x_core_group_ap_2
  - ⊟ Peer Core Group y_core_group_1
    - ⊞ Cell cell_y, Core Group Access Point y_core_group_ap_1
  - ⊟ Peer Core Group z_core_group_1
    - Proxy peer core group y_core_group_1

Figure 15. Example proxy peer configuration

By completing this task, you enabled one-way communication between the *core_group_x* and *core_group_z* core groups. If you want to configure communication both ways, you must repeat these steps, configuring a peer access point in the *core_group_z* core group that contains a proxy peer.

Core group bridge custom properties

Use these custom properties for advanced configurations for core groups and core groups that communicate with the core group bridge.

CGB_ENABLE_602_FEATURES

You can define the CGB_ENABLE_602_FEATURES custom property on all of the access point groups in your configuration if you want to be able to add core group bridge servers to the configuration without restarting the other servers in the configuration. After you enable this property, you can add a core group bridge server in one cell, without modifying the configuration in the other cell to include peer ports for the core group bridge server. Instead of manually configuring peer ports in each of the cells, you can configure the peer ports in one cell and let the other cell discover the peer ports for the first cell.

The existence of the CGB_ENABLE_602_FEATURES property enables the property. Therefore, you can set the value to any string value. Setting the value to false does not disable the property. To disable the property, you must remove it from the list of defined custom properties or change its name.

For more information about enabling this property, see “Configuring the core group bridge between core groups that are in different cells” on page 497.

FW_PASSIVE_MEMBER

Use this property in a core group bridge configuration when there is a firewall between the core groups and the secure side of the firewall is configured to listen only.

Set the FW_PASSIVE_MEMBER custom property to make the bridge interfaces that are in the core group access point passive. Set the value on the core group access point that is on the secure side of the firewall so that the core group bridge interfaces listen for connections from the unsecured side of the firewall but do not initiate any connections. The servers on the secure side of the firewall are passive. The custom property should correspond to your defined firewall rules that allow connections from the unsecured region to the secure region only.

To configure this custom property, click **Servers** > **Core groups** > **Core group bridge settings** > **Access point groups** > *access_point_group_name* > **Core group access points** > *core_group_access_point_name* > **Show detail** > **Custom properties** > **New** in the administrative console.

You also must set this custom property in any peer access points that refer to the core group access points that you configure with this custom property.

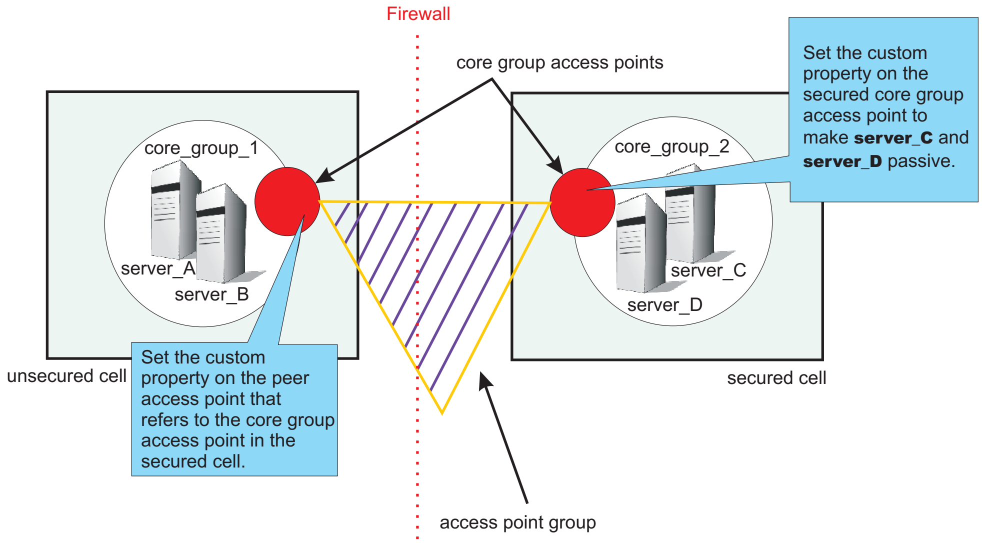

Firewall
core group access points
Set the custom property on the secured core group access point to make **server_C** and **server_D** passive.
core_group_1
server_A
server_B
core_group_2
server_C
server_D
unsecured cell
Set the custom property on the peer access point that refers to the core group access point in the secured cell.
secured cell
access point group

Figure 16. Configuring the FW_PASSIVE_MEMBER custom property

For example, *server_A* and *server_B* are configured in *core_group_1*. *Server_C* and *server_D* are configured in *core_group_2*. *Core_group_2* is behind a firewall that is configured to listen only through the firewall. *Core_group_1* is on the unsecured side of the firewall. *Core_group_1* and *core_group_2* can communicate with each other through an access point group. To configure *server_C* and *server_D* to be passive, perform the following steps:

1. In the administrative console for the cell that contains *core_group_2*, click **Servers** > **Core groups** > **Core group bridge settings** > **Access point groups** > *access_point_group_name* > **Core group access points** > *core_group_access_point_name* > **Show detail** > **Custom properties** >**New** .
2. Add the FW_PASSIVE_MEMBER custom property. Enter any value to enable the property.
3. In the administrative console for the cell that contains *core_group_1*, click **Servers** > **Core groups** > **Core group bridge settings** > **Access point groups** > *access_point_group_name* > **Peer access points** > *peer_access_point_name* > **Show detail** > **Custom properties** > **New**. The peer access point you select should correspond to the core group access point for *core_group_2*.
4. Add the FW_PASSIVE_MEMBER custom property. Enter any value to enable the property.

By configuring the FW_PASSIVE_MEMBER custom property, you configured the servers on the secured side of the firewall, *server_C* and *server_D*, to be passive. These servers listen for connections from the other side of the firewall but do not initiate any connections to the unsecured side of the firewall.

IBM_CS_LS_DATASTACK_MEG

Use this custom property to eliminate a condition that is reported by a message that is displayed repeatedly in your SystemOut.log file.

You might see a message similar to the following message in the SystemOut.log file multiple times:

```
[9/24/04 13:28:19:497 CDT] 00000013 VSync         W
DCSV2005W: DCS Stack DefaultAccessPointGroup.P at Member 172.16.1.86:9353:
The amount of memory available for synchronization is low. The configured memory
size is 418816 bytes. Currently used memory size is 420307 bytes.
```

If the member IP address is in the format of a dotted decimal IP address and port, you can eliminate these messages by increasing the amount of memory that is allocated to the core stack that is used for core group communication. Increase the value of this property until you no longer see the message in your SystemOut.log file. Because the memory is dynamically allocated, setting a larger stack size than you need does not cause memory problems.

Set the custom property on the bridge interface that contains the particular member that is in the messages. You can also set the custom property on the access point group or the core group access point. If you set the value on the access point group or core group access point, all the bridge interfaces that are in the particular group are affected. If you set the value on an individual bridge interface and an access point group or core group access point, the value that is set for the bridge interface is used. If the value is set on both an access point group and a core group access point, the value that is set for the core group access point is used.

To configure this custom property, complete the following steps:

1. Set the custom property in the administrative console.
   - To set the custom property on a bridge interface, click **Servers** > **Core groups** > **Core group bridge settings** > **Access point groups** > *access_point_group_name* > **Core group access points** > *core_group_access_point_name* > **Show detail** > **Bridge interfaces** > *bridge_interface_name* > **Custom properties** > **New**.
   - To set the custom property on a core group access point, click **Servers** > **Core groups** > **Core group bridge settings** > **Access point groups** > *access_point_group_name* > **Core group access points** > *core_group_access_point_name* > **Show detail** > **Custom properties** > **New**.
   - To set the custom property on an access point group, click **Servers** > **Core groups** > **Core group bridge settings** > **Access point groups** > *access_point_group_name* > **Custom properties** > **New**.
2. Add the IBM_CS_LS_DATASTACK_MEG custom property. Enter a value that is greater than the default value of 5 megabytes.

Units	megabytes
Default	5

Application update procedure in a high availability environment

Application update involves distributing new application binaries to each of the servers in a cluster during configuration synchronization.

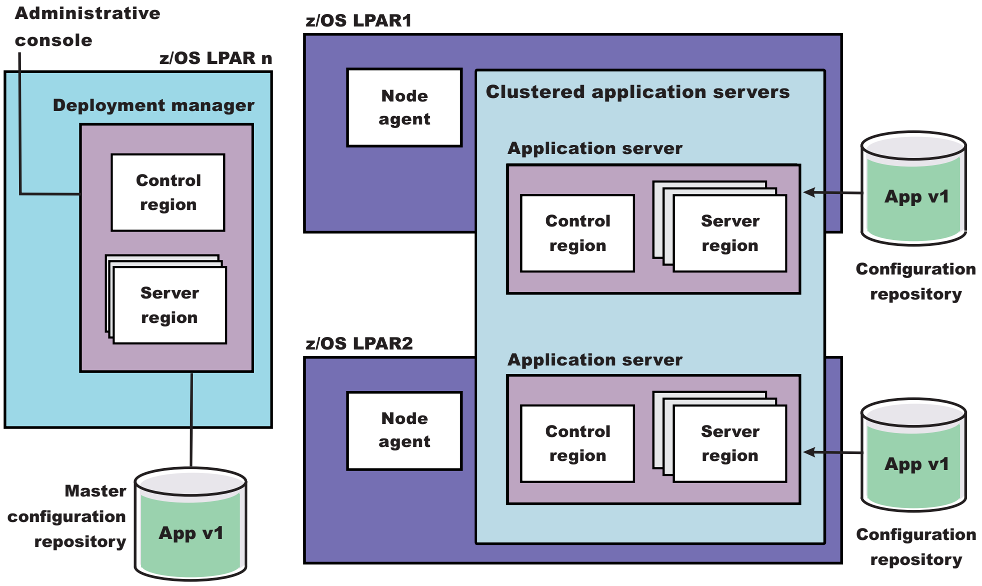


Figure 17. Application steady state - configuration view. This figure illustrates the configuration view of an application in steady state.

This distribution occurs via HTTP and happens even if the servers and the deployment manager are all located on the same LPAR. If the **synchronize with nodes** option is checked when the application update is saved, the synchronization request is sent to each node.

Normally, upon receiving the request, each node asynchronously orchestrates the synchronization process with the deployment manager. During this synchronization process, the application's binaries are downloaded from the deployment manager and stored on the node in the designated location (for example, installedApps).

The act of storing the new binaries triggers a configuration change event listener which then stops and restarts the application. Depending on such variables as dispatcher behavior, LPAR weighting, etc., a variance in the order and pace at which each server makes the new application available may be observed.

Because of the concurrent, asynchronous nature of node synchronization, continuous availability of the application being updated is not guaranteed. This is because there is no correspondence between the actual state of the application and the workload routing mechanisms. Client requests may be routed to a server even if that particular application is temporarily unavailable.

In a high availability environment, an application must remain available even during an update process. Therefore, application rollout to each cluster member, as well as workload routing to the cluster members must be carefully controlled to prevent workload from being routed to a cluster member that is undergoing the update process. When these two aspects are carefully controlled, an update can be installed on each cluster member without client requests arriving at a cluster member during the update process.

Stopping an application server to manually update a high availability application

For manual application rollout, workload routing is controlled by stopping the application server on which the cluster member being updated resides. This results in a quiesce of that server. All existing requests already in the server are allowed to complete, but no new requests are accepted. Both the sysplex distributor and the WebSphere Application Server Web server plug-in routes work away from the quiescing server. After all work has completed, you start the application update process on this server.

Determine which application servers are hosting the cluster members that need updating.

If you have a high availability application whose updates you want to manually control you can use this process or you can use the MVS Modify command to pause the listeners for the affected application servers.

To manually control application rollout and workload routing in a high availability environment:

1. Disable all forms of automatic synchronization, across all nodes in the cell and save the changes. Perform one of the following processes to complete this step:
   - In the administrative console:
     a. Click **System administration** > **Node agents** > *node_agent_name* > **File Synchronization Service**.
     b. Unselect the **Automatic Synchronization** and **Startup Synchronization** options.
     c. Select the **Synchronize changes with nodes** option.
     d. Click **Save**.
   - Use wsadmin scripting to specify the following commands and then restart all affected node agents:

```
set node NODE
set na_id [$AdminConfig getid /Node:$node/Server:nodeagent/]
set syncServ [$AdminConfig list ConfigSynchronizationService $na_id]
$AdminConfig modify $syncServ {{autoSynchEnabled false}}
$AdminConfig modify $syncServ {{synchOnServerStartup false}}
$AdminConfig save

set nodeSync [$AdminControl completeObjectName type=NodeSync,node=$node,*]
$AdminControl invoke $nodeSync sync
```

   Important: For a production environment, it is reasonable to always run the node agent with automatic synchronization disabled. However, it is advisable for startup synchronization to be enabled for the node agent so that it can acquire configuration updates that occur when the node agent is down. Startup Synchronization can be left enabled provided you can ensure that you will not restart the node agent manually, through automation, or through automatic restart manager during the application update process.

2. Update the application in the master configuration repository on the deployment manager server. Perform one of the following processes to complete this step:
   - In the administrative console:
     a. Click **Applications** > **Enterprise Applications**.
     b. Select the application you want to update.
     c. Complete the application update process.
     d. Save your changes to the master configuration. **DO NOT** select the **Synchronize changes with nodes** option .
   - Use wsadmin scripting to issue the following command:

```
set app_loc /path/to/app
set app_options {application options ie: -appname app}
set options [list -update]  lappend options $app_options
$AdminApp install $app_loc $options
$AdminConfig save
```

At this point, you have the updated the version of your application (App v2 in the following figure) in your Master Configuration. However, the original version of your application (App v1 in the following figure) is still running in the cluster that has Cluster members on LPAR1 and LPAR2.

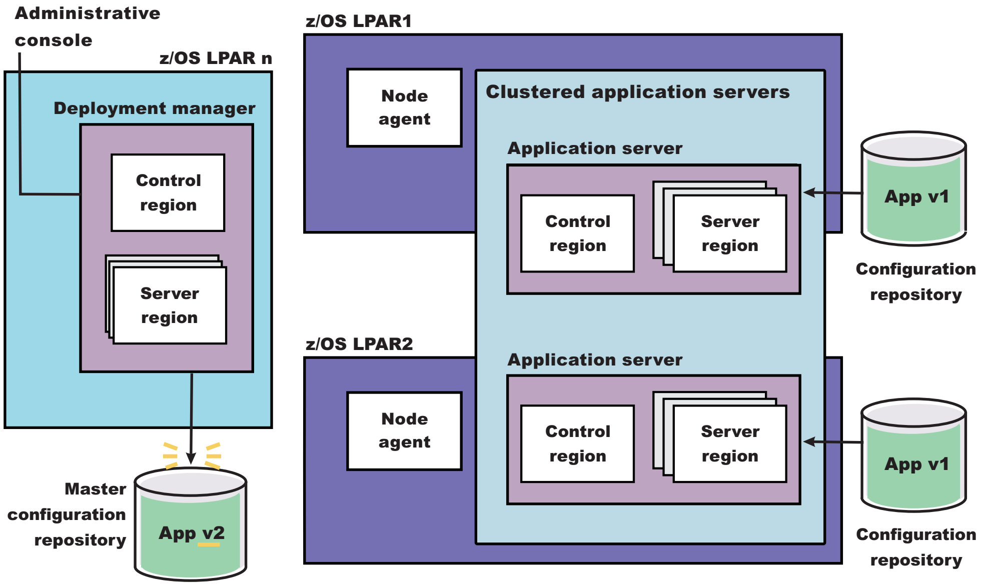


Figure 18. Install application update. This figure illustrates the first stage of an application update in a high availability environment.

3. Stop the Application Server on LPAR1 and manually synchronize the node to the updated version of the application. This step may take time to complete because the server must wait for all currently assigned work items to complete before shutting down.

   Perform one of the following processes to complete this step:

   - In the administrative console:
     a. Click **Servers** > **Application Servers** .
     b. Select the cluster member you want to stop and update. This cluster member should be on LPAR1.
     c. Click **STOP**. The Cluster Stop method should not be used, because it will stop all Servers within the cluster and the application will no longer be available.
   - Use wsadmin scripting: to issue the following commands:

     ```
     set node NODE
     set server SERVER
     $AdminControl stopServer $server $node
     ```
   - Issue the following command from the MVS Console:

     ```
      STOP short_server_name
     ```

     For example:

     ```
     STOP BBOS001
     ```

4. Synchronize the node. Perform one of the following processes to complete this step:
   - In the administrative console:
     a. Click **System Administration** > **Nodes**.
     b. Select the node you want to synchronize, and then click **Full Resynchronize**.
   - Use wsadmin scripting to issue the following commands:

     ```
     set node NODE
     set nodeSync [$AdminControl completeObjectName type=NodeSync,node=$node,*]
     $AdminControl invoke $nodeSync sync
     ```

   As illustrated in the following figure, the updated version of the application (App v2) now resides in the node on LPAR1.

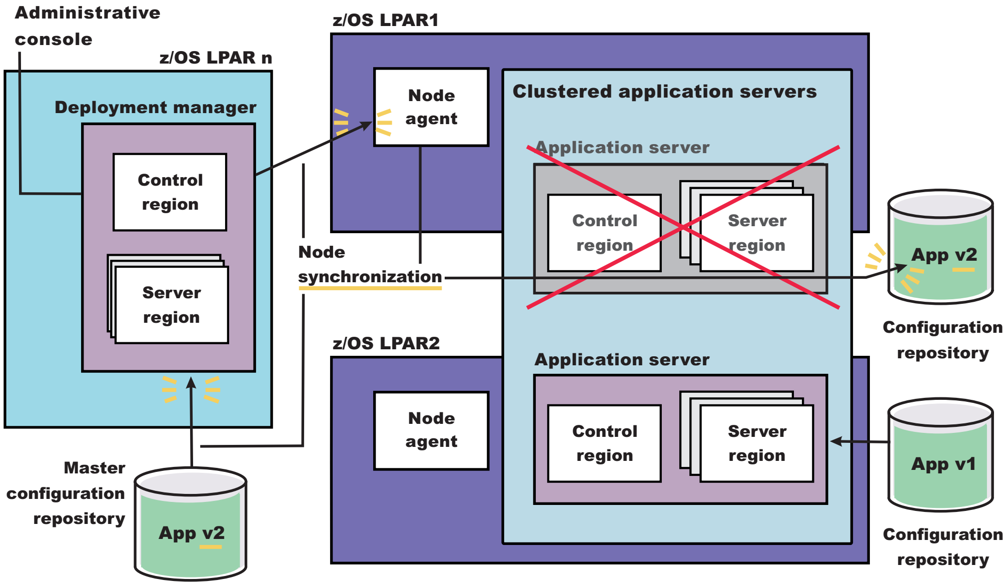


Figure 19. Update the node on LPAR1. This figure illustrates the first stage of an application update in a high availability environment with two LPARs.

5. Restart the server on LPAR1. Perform one of the following processes to complete this step:
   - In the administrative console:
     a. Click **Servers** > **Application servers**.
     b. Select the server you want to start, and then click **START**.
   - Use wsadmin scripting to issue the following commands:

     ```
     set node NODE
     set server SERVER
     $AdminControl startServer $server $node
     ```

   - Issue the following command from the MVS Console:

     START *procname*,JOBNAME=*server_short_name*.ENV=*cell_short_name*.*node_short_name*.*server_short_name*

     For example:

     ```
     START BBO6ACR,JOBNAME=BBOS001,ENV=PLEX1.SY1.BBOS001
     ```

   When this server comes back up, it will be running the new version of the application (App v2),

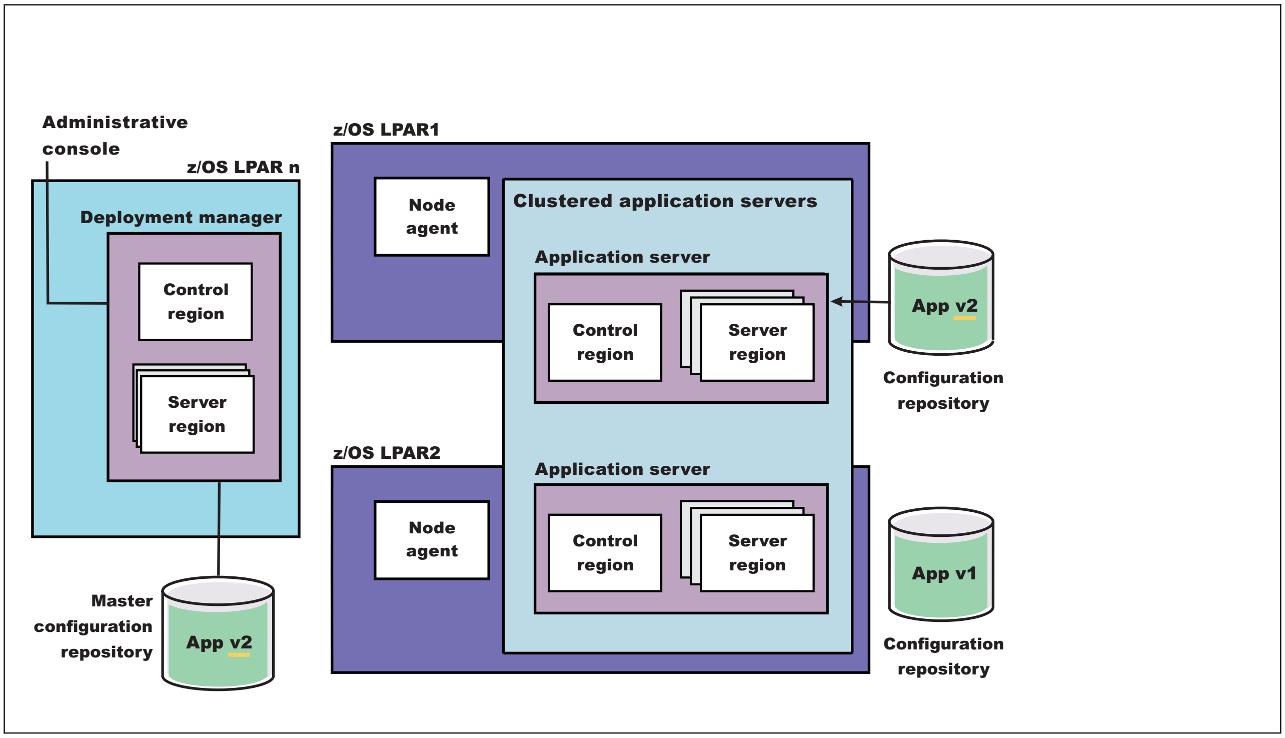


Figure 20. Restart the server on LPAR1. This figure illustrates the completion of the first stage of an application update in a high availability environment.

6. With the new version of the application running on LPAR1, repeat the preceding three steps on the other LPARs in the cluster to update them with the new version of the application. The following figure illustrates what your configuration will look like in a two LPAR cluster.

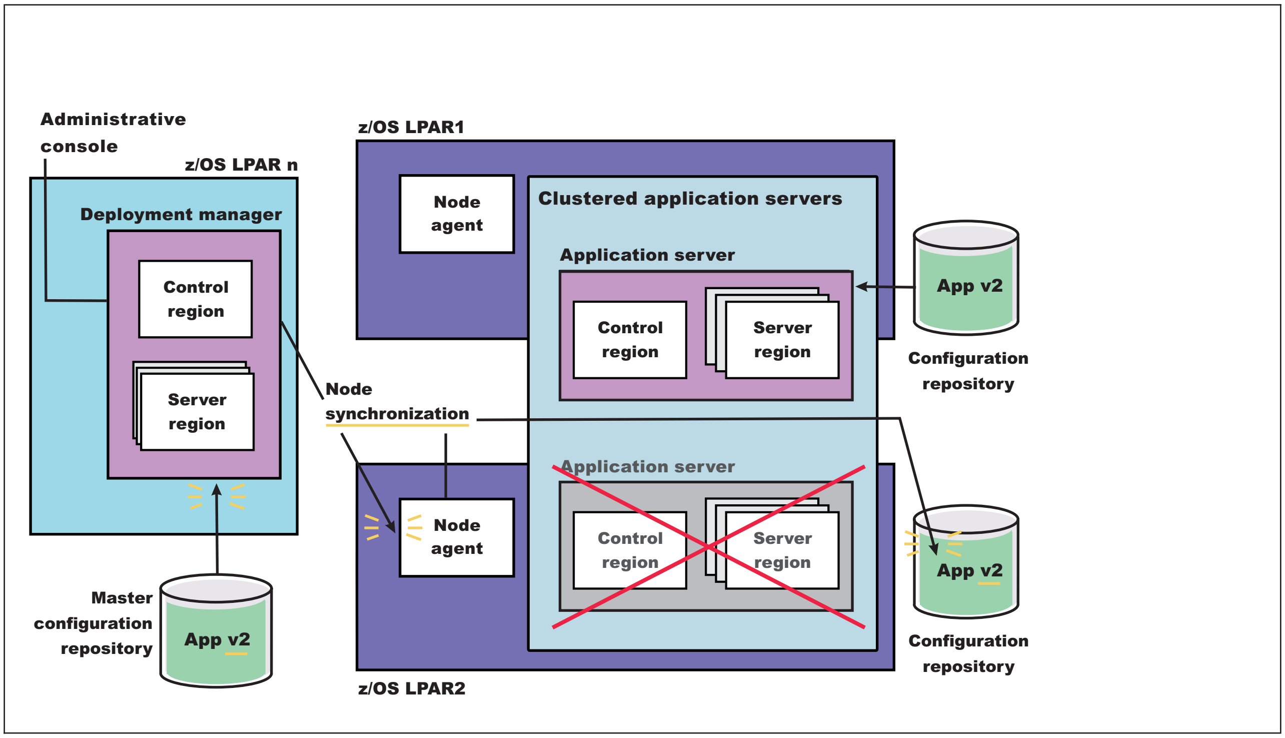


Figure 21. Update the node on LPAR2. This figure illustrates the second stage of an application update in a high availability environment.

The application update process is complete when the new version of the application is running on all of the cluster members in the cluster. The following figure illustrates what a two LPAR cluster will look like after you restart the server on LPAR2.

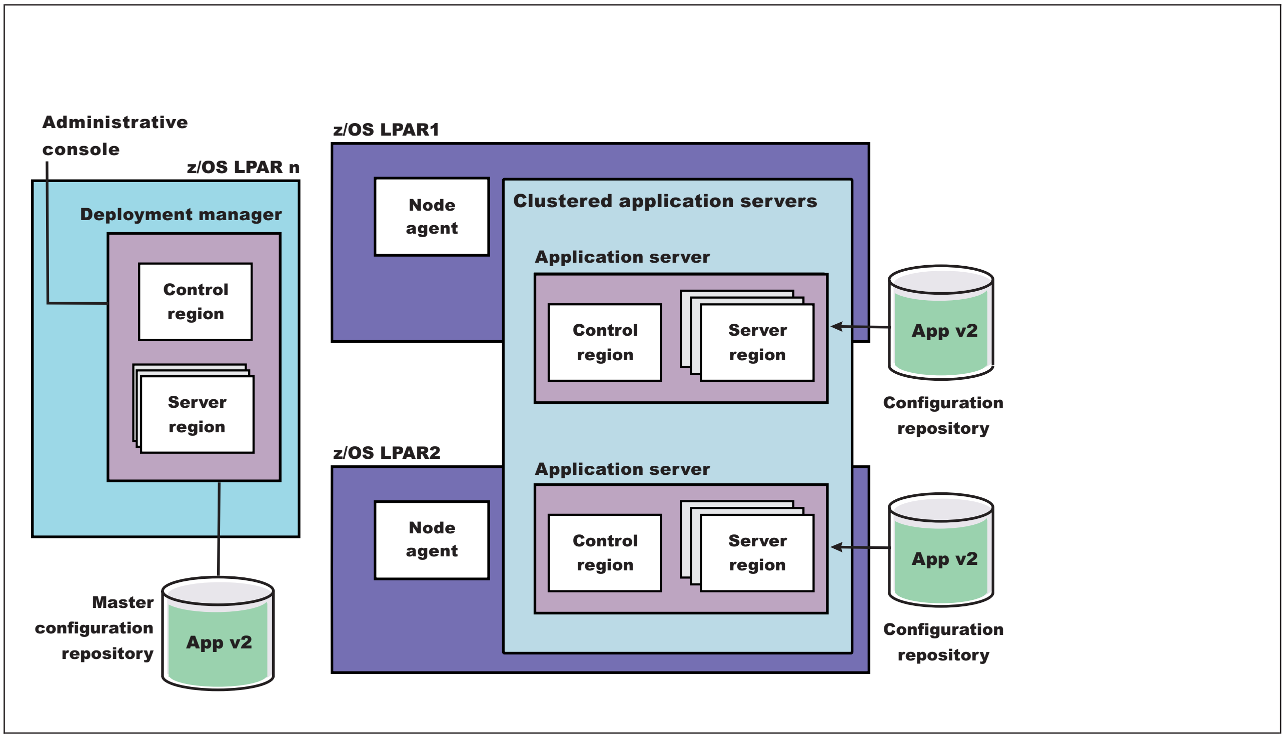


Figure 22. Restart server on LPAR2. This figure illustrates what a two LPAR cluster will look like after you restart the server on LPAR2.

Pausing an application server listener to manually update a high availability application

Instead of stopping the application server, you can use the MVS console Modify command to pause the listeners for that application server, perform the application update, and then resume the listeners. If you use this technique, you do not have to stop and then start the server to perform the application update.

Determine which application servers are hosting the cluster members that need updating.

If you have a high availability application whose updates you want to manually control, but you do not want to stop the affected servers, you can use the MVS Modify command to pause the listener for each of these servers and then update the application.

To pause the listeners and manually control application rollout in a high availability environment:

Note:

1. Disable all forms of automatic synchronization, across all nodes in the cell and save the changes. Perform one of the following processes to complete this step:
   - In the administrative console:
     a. Click **System administration** > **Node agents** > *node_agent_name* > **File Synchronization Service**.
     b. Unselect the **Automatic Synchronization** and **Startup Synchronization** options.
     c. Select the **Synchronize changes with nodes** option.
     d. Click **Save**.

- Use wsadmin scripting to specify the following commands and then restart all affected node agents:

  ```
  set node NODE
  set na_id [$AdminConfig getid /Node:$node/Server:nodeagent/]
  set syncServ [$AdminConfig list ConfigSynchronizationService $na_id]
  $AdminConfig modify $syncServ {{autoSynchEnabled false}}
  $AdminConfig modify $syncServ {{synchOnServerStartup false}}
  $AdminConfig save

  set nodeSync [$AdminControl completeObjectName type=NodeSync,node=$node,*]
  $AdminControl invoke $nodeSync sync
  ```

  Note: For a production environment, it is reasonable to always run the node agent with automatic synchronization disabled. However, it is advisable for startup synchronization to be enabled for the node agent so that it can acquire configuration updates that occur when the node agent is down. Startup Synchronization can be left enabled provided you can ensure that you will not restart the node agent manually, through automation, or through automatic restart manager during the application update process.

2. Update the application in the master configuration repository on the deployment manager server. Perform one of the following processes to complete this step:

   - In the administrative console:

     a. Click **Applications** > **Enterprise Applications**.

     b. Select the application you want to update.

     c. Complete the application update process.

     d. Save your changes to the master configuration. **DO NOT** select the **Synchronize changes with nodes** option .

   - Use wsadmin scripting to issue the following command:

     ```
     set app_loc /path/to/app
     set app_options {application options ie: -appname app}
     set options [list -update]  lappend options $app_options
     $AdminApp install $app_loc $options
     $AdminConfig save
     ```

     At this point, you have the updated the version of your application (App v2 in the following figure) in your Master Configuration. However, the original version of your application (App v1 in the following figure) is still running in the cluster that has Cluster members on LPAR1 and LPAR2.

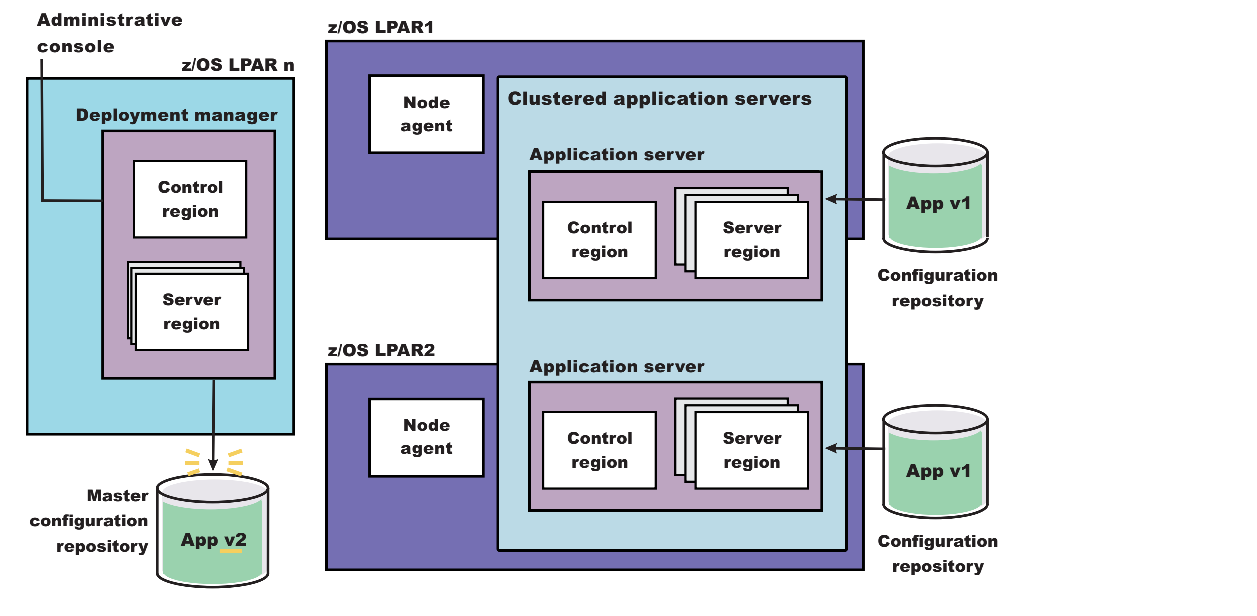


Figure 23. Install application update. This figure illustrates the first stage of an application update in a high availability environment.

3. Pause the listener of the application server on LPAR1 and manually synchronize the node to the updated version of the application. After you pause the listener, wait for all work items currently assigned to the server to complete, and then issue the following command from the MVS Console:

   ```
   MODIFY short_server_name,PAUSELISTENERS
   ```

   For example, if the short name for the server you are pausing is BBOS001, issue the following command:

   ```
   MODIFY BBOS001,PAUSELISTENERS
   ```

4. Synchronize the node. Perform one of the following processes to complete this step:
   - In the administrative console:
     a. Click **System Administration** > **Nodes**.
     b. Select the node you want to synchronize, and then click **Full Resynchronize**.
   - Use wsadmin scripting to issue the following commands:

     ```
     set node NODE
     set nodeSync [$AdminControl completeObjectName type=NodeSync,node=$node,*]
     $AdminControl invoke $nodeSync sync
     ```

   As illustrated in the following figure, the updated version of the application (App v2) now resides in the node on LPAR1.

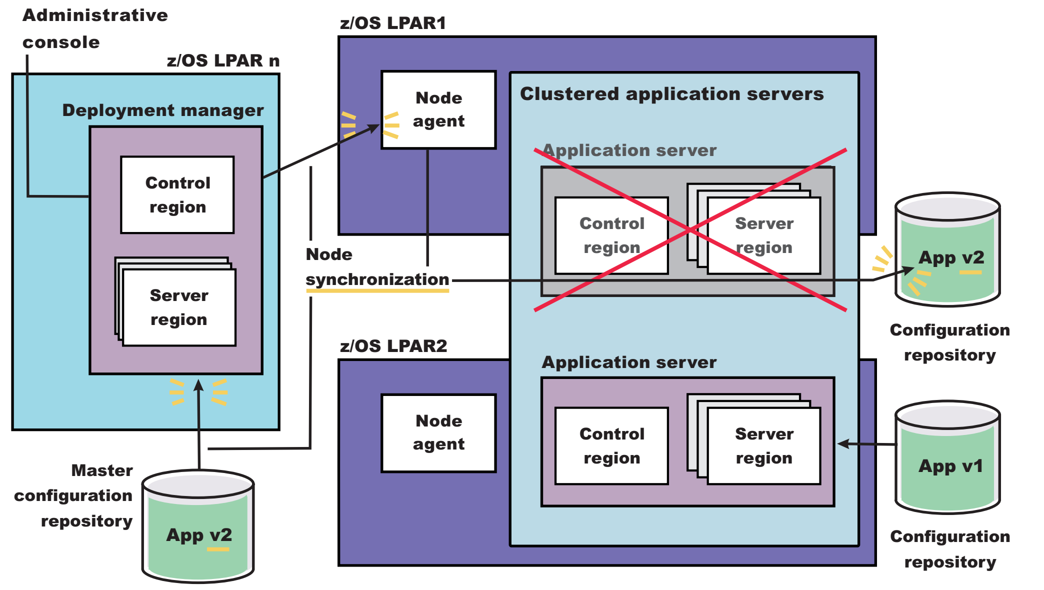


Figure 24. Update the node on LPAR1. This figure illustrates the first stage of an application update in a high availability environment with two LPARs.

5. Resume the listener of the application server on LPAR1. Issue the following command from the MVS Console:

   ```
   MODIFY short_server_name,RESUMELISTENERS
   ```

   For example, if the short name for the server you are pausing is BBOS001, issue the following command:

   ```
   MODIFY BBOS001,RESUMELISTENERS
   ```

6. With the new version of the application running on LPAR1, repeat the preceding three steps on the other LPARs in the cluster to update them with the new version of the application. The following figure illustrates what your configuration will look like in a two LPAR cluster.

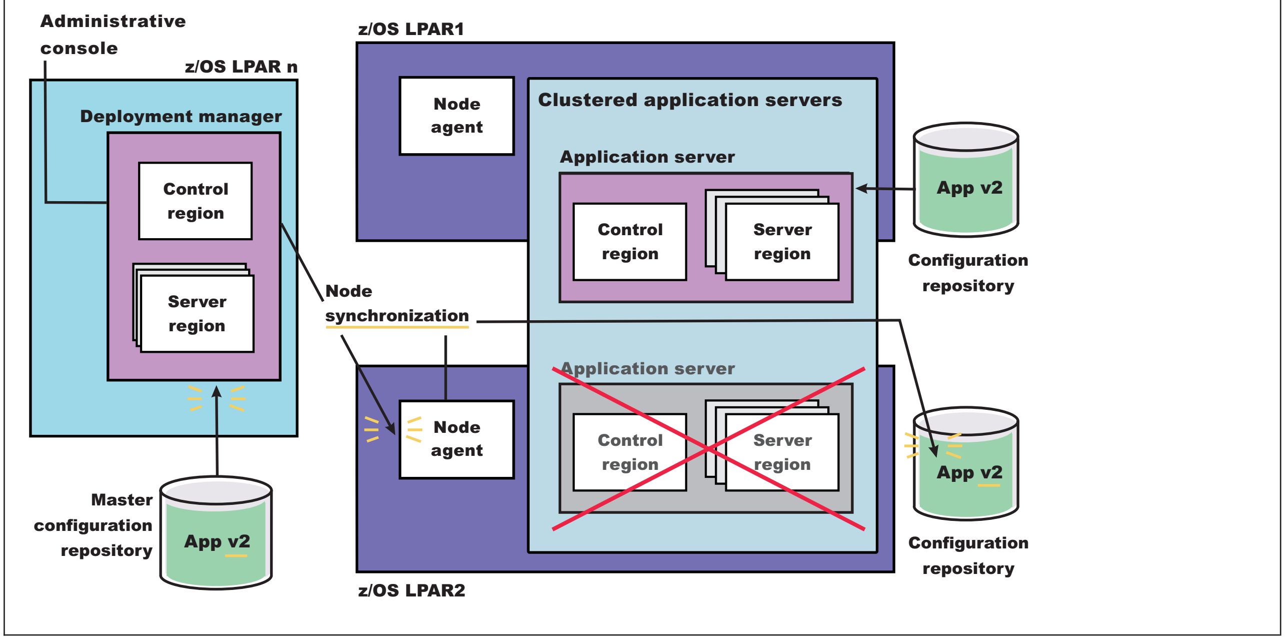


Figure 25. Update the node on LPAR2. This figure illustrates the second stage of an application update in a high availability environment.

The application update process is complete when the new version of the application is running on all of the LPARs in the cluster.

Automatically rolling out updates to a high availability application

You can set up your system to perform automatic application rollout for your high availability applications. The automatic application rollout update process stops or pauses each application server that is hosting a cluster member that needs updating.

Determine which application servers are hosting the cluster members that need updating.

If you have a high availability application that frequently requires updates, you might want to automatically control the rollout of these updates.

When setting up the rollout update process you must decide whether you want the application servers to stop or pause while the update is made to an application. If you want the servers to pause, you must configure the node agent to allow the rollout update process to pause and resume the servers. You do not have to make any configuration changes if you want the rollout update process to stop and start the servers. However if the rollout update process stops and starts the servers, the process takes much longer to complete.

When an application server pauses, all requests already in the queue for that server are allowed to complete, but no new requests are accepted. Both the sysplex distributor and the WebSphere Application Server Web server plug-in route work away from the server that is paused. After all of the requests assigned to that server complete, the application update process starts on that server.

When the update process is complete, the listener for that server resumes and the sysplex distributor and the WebSphere Application Server Web server plug-in assign new work to that server. This process is repeated for all of the other servers in the cluster until all of the affected cluster members are updated.

To prepare your system to automatically rollout updates to a high availability application:

1. Determine whether you want the rollout update process to stop or pause the affected application servers.
   - If you want the rollout update process to stop a server before it performs an application update, go to step 5.
   - If you want the rollout process to pause a server before it performs an application update, go to step 2. Steps 2, 3, and 4 are configuration changes that enable the rollout update process to pause and resume servers during an application update. You only have to make these changes once.
2. Add the **com.ibm.websphere.zos.mvsservices.enable** and **com.ibm.websphere.zos.rollout.pauseresume** custom properties to the node agent settings in the master configuration repository on the deployment manager server. These properties must be added to the settings for all of the node agents on which you want to automatically start the MVSServices MBean.
   a. In the administrative console, click **System administration > Node agents >** *node_agent_name* **> Administration Services > Custom Properties > New**
   b. Enter **com.ibm.websphere.zos.mvsservices.enable** in the Name field, and **true** in the Value field.
   c. Click **Ok** .
   d. Click **New** .
   e. Enter **com.ibm.websphere.zos.rollout.pauseresume** in the Name field, and **true** in the Value field.
   f. Click **Ok** .
   g. Repeat these steps for any other node agents on on which you want to automatically start the MVSServices MBean.
3. Click Save to save your changes directly to the master configuration.

   After you add the com.ibm.websphere.zos.rollout.pauseresume custom property and set it to `true`, any future application rollouts on this node are accomplished by pausing a listener for the application server, rather than stopping that application server.

   If the custom property com.ibm.websphere.zos.rollout.pauseresume is set to true, but the MVSServices MBean is not running on the configured node, the application servers on that node do not pause, and are not updated during the application update process.

   Messages are displayed on the MVS Console when an application server is paused or resumed similar to the messages that are displayed when an application server is stopped or started.
4. Restart the node agent. When you restart the node agent, the MVSServices MBean automatically starts.
5. Update the application configuration repository in the master on the deployment manager server. See the *Administering applications and their environment* PDF for more information.

You are ready to start the rollout update process for an application that you need to update.

To start the rollout update process, in the administrative console, click **Applications > Enterprise applications**, select the application to update, and click **Rollout Update**.

Important: The application you select must reside on at least one member of a cluster.

This function automatically stops or pauses the server, updates the application, and then starts or resumes the server. Nodes are processed one at a time, so only the server residing on the node being processed is affected, the servers on the other nodes continue to process work. Eventually all of the nodes and servers are updated.

The update process is complete when the updated version of the application is running on all of the LPARs in the cluster.

Setting up a high availability sysplex environment

Setting up a high availability sysplex environment enables you to control application rollout and workload routing.

- The high availability sysplex must include at least two logical partitions (LPARs). These LPARs should be on separate hardware instances to eliminate hardware single points of failure (SPOFs).
- There must be a network path redundancy leading up to the Web servers and Applications Servers in your sysplex.
- If you are using HTTP sessions, session state must be shared between cluster member using the data replication service (DRS), or your session data must be stored in DB2. If you are using stateful session Enterprise JavaBeans (EJBs), the stateful session persistent store must be configured on a shared HFS. It is not recommended that you use stateful session Enterprise JavaBeans.

To set up a WebSphere Application Server for z/OS high availability sysplex environment:

1. Configure a WebSphere Application Server for z/OS node on each LPAR that is configured in the Network Deployment cell. The deployment manager Server, which is required, must be configured on its own node. It can be configured on either LPAR or on a separate LPAR.
2. Use the Administrative Console to make sure a location service daemon has been defined on each LPAR that has one or more nodes in the same cell. “Modifying z/OS location service daemon settings” on page 196 describes how to define a location service daemon if one is not already defined.
3. Define an application server on each node, and form all of the application servers into a cluster.

   See“Adding members to a cluster” on page 409 for more information on how to add application servers to a cluster.
4. Define the following dynamic virtual IP addresses (DVIPAs) through the z/OS Operating System’s Sysplex Distributor.
   - A dynamic virtual IP address as the daemon’s IP name for the cell. This IP address enables WLM-balanced workload routing and fail over between the LPARs for IIOP requests.
   - A dynamic virtual IP address as the HTTP transport channel name for the cell. This IP address enables WLM-balanced routing and fail over between the LPARs for sessionless HTTP requests.

   See the *z/OS Communications Server IP Configuration Guide* for your version of the z/OS operating system for a description of how to define IP addresses through the z/OS Sysplex Distributor. This publication is available at http://www.ibm.com/servers/eserver/zseries/zos/bkserv/v1r4books.html.
5. Define a static IP address for each node as an auxiliary HTTP transport channel name for the cell. This IP address enables directed HTTP routing for sessional HTTP requests.
6. Configure Web server plug-ins in each of the Web servers. Configure the plug-ins to use the HTTP DVIPA for sessionless requests and the static IP addresses for sessional requests. See Chapter 3, “Communicating with Web servers,” on page 117 for more information.

High availability configuration

The objective of any high availability configuration is to eliminate all single points of failure (SPOFs).

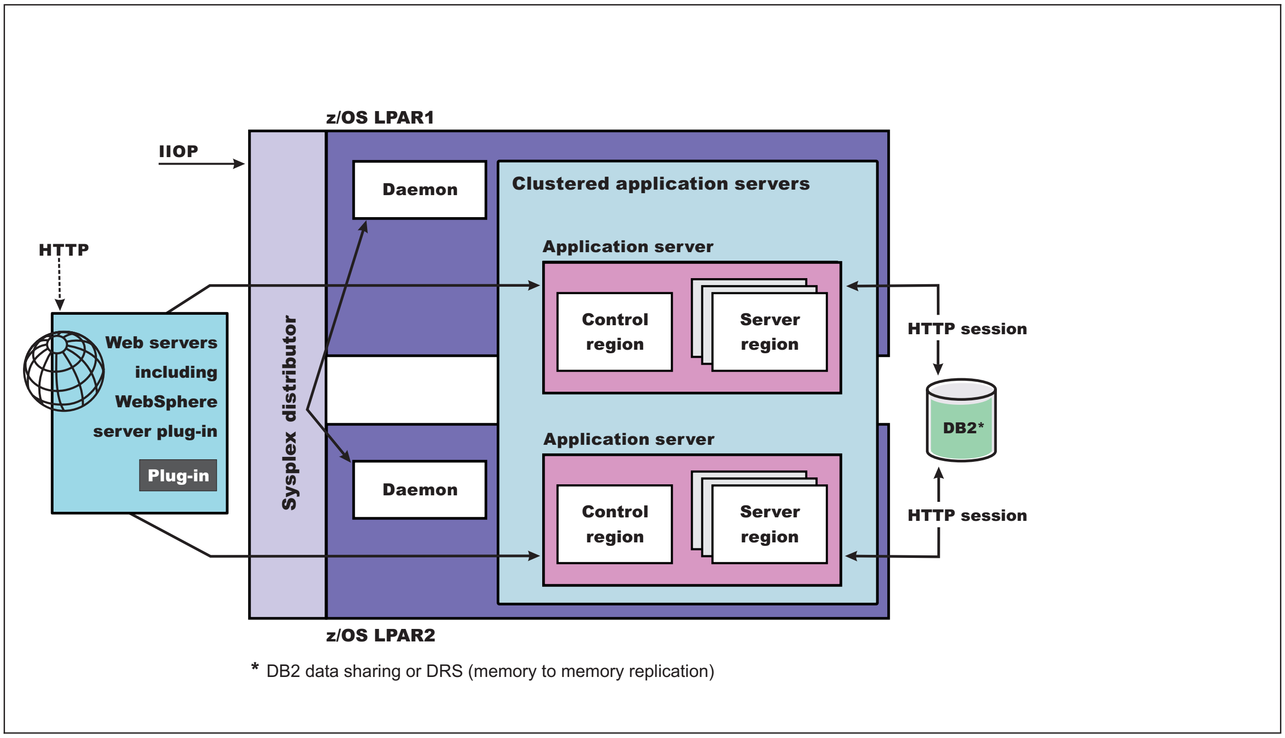


Figure 26. WebSphere Application Server for z/OS High Availability Configuration. This figure illustrates the recommended WebSphere Application Server for z/OS configuration for high availability. The key elements are described in the text that accompanies this figure.

Following are the key elements of a WebSphere Application Server for z/OS high availability configuration:

- Network path redundancy leading up to the Web servers and Applications Servers.
- Redundant Web servers. (There must be at least two logical partitions (LPARs) in a high availability sysplex configuration.)
- A highly available sysplex configuration. These LPARs should be on separate hardware instances to eliminate hardware and software Single Points of Failures (SPOFs).
- A WebSphere Application Server for z/OS node on each LPAR that is configured into a Network Deployment cell. The deployment manager server (required, and configured on its own node) can be configured on either LPAR or on a separate LPAR. (The deployment manager server is not depicted in the preceding figure.) Also note, there is a daemon process (WebSphere CORBA Location Service) on each LPAR that has one or more nodes in the same cell.
- An application server defined on each node, and formed into a server cluster with the other application servers in the network.
- A dynamic virtual IP address (DVIPA) defined through the z/OS Sysplex Distributor as the daemon IP name for the cell. This IP address enables WLM-balanced routing and fail over between the LPARs for IIOP requests.
- A dynamic virtual IP address (DVIPA) defined through Sysplex Distributor as the HTTP transport name for the cell. This IP address enables WLM-balanced routing and fail over between the LPARs for sessionless HTTP requests.
- A static IP address is required for each node as an auxiliary HTTP transport name for the cell. This enables directed HTTP routing for sessional HTTP requests.

- A WebSphere Web server plug-in must be installed in each of the Web servers and configured to use the HTTP DVIPA for sessionless requests, and the static IP addresses for sessional requests.
- If using HTTP sessions, session state must be shared between cluster member using the data replication service (DRS) or session data must be stored in DB2. If you are using stateful session Enterprise JavaBeans, the stateful session persistent store must be configured on a shared HFS. (Using stateful session Enterprise JavaBeans is not a best practice.)

Troubleshooting high availability environment problems

Review the following topics if you encounter a problem with your high availability environment.

Message HMGR0218I is not displayed after a Java Virtual Machine starts

In a properly set up high availability environment, a high availability manager can reassess the environment it is managing and accept new components as they are added to the environment. For example, when a Java virtual machine (JVM) is added to the infrastructure, a discovery process begins. During startup the JVM tries to contact the other members of the core group. When it finds another running JVM, it initiates a join process with that JVM that determines whether or not the JVM can join the core group. If the new JVM is accepted as a member of the core group, all of the JVMs, including the new one, log message HMGR0218I . This message is also displayed on the administrative console.

Message HMGR0218I indicates the number of application servers in the core group that are currently online. If this message is not displayed after a JVM starts, either a configuration problem or a communication problem has occurred. To fix this situation, verify that the application server is running on a current configuration, by either using the deployment manager to tell the node agent to synchronize, or use the **syncNode** command o manually perform the synchronization. If the JVM still cannot join the core group, a network configuration problem exists.

CPU starvation messages in SystemOut.log

CPU starvation detected error messages are displayed in the SystemOut.log file whenever there is not enough physical memory available to allow the high availability manager threads to have consistent runtimes. When the CPU is spending the majority of its time trying to load swapped-out processes while processing incoming work, thread starvation might occur. The high availability manager detects this condition, and logs these error messages informing you that threads are not getting the required runtime.

To achieve good performance and avoid receiving these error messages, it is recommended that you allocate at least 512 MB of RAM for each Java process running on a single machine.

High CPU usage in a large cell configuration when security is enabled

With certain configurations and states, the amount of time spent in discovery becomes substantial.

- If a large the number of processes are defined within a core group, a proportionally large number of connections must be established to support these processes.
- If a large number of inactive processes are defined within a core group, a proportionally large number of connections are attempted during each discovery interval.
- If administrative security is enabled, the DCS connections are secured, and the impact of opening a connection greatly increases .

To decrease the CPU time spent in discovery:

1. In the administrative console, click **Servers > Core groups > Core groups settings** , and then select the -> **DefaultCoreGroup**.
2. Under Additional Properties, click **Custom properties > New**.
3. Enter `IBM_CS_UNICAST_DISCOVERY_INTERVAL_SECS` in the Name field and 120 in the Value field.

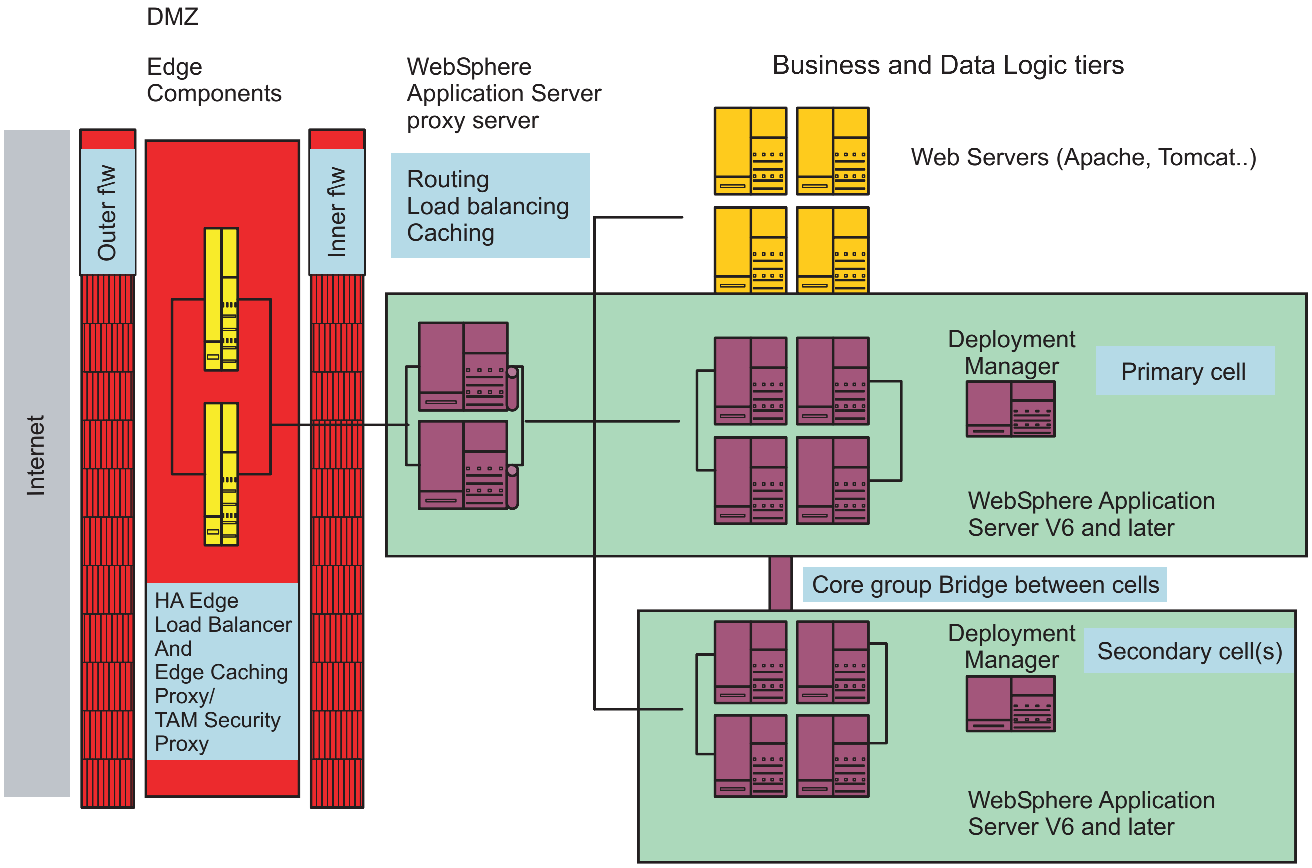
4. Click **OK**.
5. Then click **New** again and enter `IBM_CS_SS_SECURE_TOKEN` in the Name field and`false` in the Value field.
6. Click **OK** and then **Save** to apply these changes to the master configuration.
7. Restart the server for these changes to take effect.

Chapter 10. Setting up the proxy server

The *proxy server* is a specific type of application server that routes HTTP requests to content servers that perform the work. The proxy server is the initial point of entry, after the firewall, for requests into the enterprise.

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
Internet
Outer flw
Inner flw
Routing
Load balancing
Caching
Web Servers (Apache, Tomcat..)
Deployment Manager
Primary cell
WebSphere Application Server V6 and later
HA Edge Load Balancer And Edge Caching Proxy/ TAM Security Proxy
Core group Bridge between cells
Deployment Manager
Secondary cell(s)
WebSphere Application Server V6 and later

In WebSphere Application Server V6.0.2, you had to augment the deployment manager profile to manage the proxy server. For WebSphere Application Server V6.1 and later versions, the proxy server is managed from the administrative console without initial augmentation.

Creating a proxy server

This topic provides information to create and configure a proxy server.

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.

1. Create a proxy server in the administrative console by clicking **Servers** > **Proxy Servers**.
2. Click **New**.
3. Select the node on which you want the proxy server to reside. Only Network Deployment nodes display in the selection list. A proxy server can reside only in a Network Deployment node. Enter a name for the new proxy server and click **Next**.
4. 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 application server.

   Tip:
   Mapping to pre-existing proxy servers 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.
5. Determine whether or not to generate unique HTTP ports by selecting or clearing Generate unique HTTP ports. Click **Next**. 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.
6. Review the summary panel and click **Finish**.

You now have a functional proxy server that automatically routes HTTP requests to WebSphere Application Server cells. To enable routing to another WebSphere Application Server cell, configure your cell to communicate with other WebSphere Application Server cells.

If the proxy server fails to start when attempting to start it as a non-privileged user on UNIX systems, change the ports of the PROXY_HTTP_ADDRESS and PROXY_HTTPS_ADDRESS transport chains to values greater than `1024`.

Migrating profiles for the WebSphere proxy server

This topic describes how V6.0.2 profiles contain the proxy server feature when migrating to a V6.1 profile.

Migrate from a V6.0.2 profile to a V6.1 profile as follows:

1. Run the **WASPreUpgrade.bat** or the **WASPreUpgrade.sh** command from the *install_root*/bin directory and point it to the V6.0.2 release. This actions makes a copy of all the required V6.0.2 files that are needed for the **WASPostUpgrade.bat** or **WASPostUpgrade.sh** command. You can delete V6.0.2, but this action is not recommended.
2. Create your corresponding profiles in V6.1, 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.

Follow the steps to disable routing to ODC-compliant application servers. The Web module proxy configuration option will only display if the deployment manager is augmented with the proxy server configuration.

1. From the administrative console, click **Applications** > **Enterprise Applications**.
2. Select the application you want to customize.
3. Click **Web modules**.
4. Select the Web module from the collection view.
5. Click **Web Module Proxy Configuration**.
6. Deselect **enable proxy** to disable routing using the proxy server.
7. Choose the protocol from the drop-down box for the Web module transport protocol field. Complete this step if you want to use a specific transport protocol, such as HTTP, for the protocol between the proxy server and the application server, rather than 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.

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** > **Enterprise Application >** *application_name* **> Manage Modules >** *Web_module_name* > **Web Module Proxy Configuration**.

Name

Specifies the name of the proxy server.

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.

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

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.

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. You can find core group bridge settings by clicking **Servers** > **Core groups** > **Core group bridge settings** in the administrative console.

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.

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. If you want to use CGB_ENABLE_602_FEATURES, enable it on all of the cells involved.
2. Restart all processes that are now bridge interfaces.

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.

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 the following tasks:

1. Creating a URI group from **Environment > URI groups** in the administrative console to define the URI patterns that are associated with the Web content that is hosted on these servers.
2. Creating a generic server cluster definition from **Servers > Generic server clusters** in the administrative console to define the endpoints that define the cluster membership.
3. Creating 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 by clicking **Servers > Proxy servers >** *server_name* **> Communications > Ports** in the administrative console.

1. Modify the ports and hosts that are associated with these endpoints by clicking **Servers > Proxy servers >** *server_name* **> Communications > Ports** in the administrative console.
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**.

Restart WebSphere Application Server for this modification to be effective.

Adding a new HTTP endpoint for the proxy server

An administrator can use the proxy server to create additional endpoints to listen for HTTP(S) requests.

The following steps will create a new transport chain:

1. Click **Servers > Proxy servers >** *<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 a custom SSL repertoire

This topic provides information to set up a custom Secure Sockets Layer (SSL) repertoire for inbound (client to proxy server) HTTPS traffic.

Follow the steps to create a new SSL repertoire that contains a valid SSL certificate:

1. In the administrative console, click **Security > SSL** to create a new SSL repertoire. See for more information on SSL repertoire settings.
2. Click **Servers > Proxy servers** *<server_name>* **> HTTP proxy server settings > Proxy server transports**.
3. Select the HTTPS transport chain to customize.
4. Click **SSL inbound channel**.
5. From the SSL repertoire drop-down menu, select the SSL repertoire that you created in step one.
6. Click **Apply**.

Caching content in the proxy server

This topic provides information on caching static and dynamic content in the proxy server.

Setting up caching in the proxy server

The proxy server provides the capability to cache and retrieve responses locally. It uses the WebSphere object cache instance as the store for the responses. Each proxy server has a default object cache instance that it uses for storage and retrieval of cached responses.

To configure the object cache instance for size, disk offload location, and other such capabilities, in the administrative console, click **Servers > Proxy servers >** *server_name* **> HTTP proxy server settings > Proxy cache instance**. Then 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. To enable caching at the proxy server, in the administrative console, click **Servers > Proxy servers >** *server_name* **> HTTP proxy server settings > Proxy settings** page in the administrative console. Then select **Enable caching** and choose a cache instance from the drop-down box.

Caching static content

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 caching is enabled by default when caching is enabled for the proxy server.

Caching dynamic content

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 WebSphere Application Server 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.

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. To enable dynamic content to be cacheable with the proxy server, in the administrative console, click **Servers > Proxy servers >** *server_name* **> HTTP proxy server settings > Proxy settings**, and then select **Cache dynamic content**. Invalidations are received by connecting to the cache update URI that is associated with the invalidation servlet hosted on the application server cluster.

Routing requests from a plug-in to a proxy server

This topic provides information on setting up a WebSphere Application Server plug-in to route requests to a proxy server.

An administrator may choose to set up a Web server, such as IBM HTTP Server, with the WebSphere Application 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:

1. From the administrative console, click **Servers > Proxy servers >** *server_name* **> HTTP proxy server settings > Proxy settings**.
2. In the **Generate plug-in configuration** drop-down menu, select the cell scope. This generates the `plugin-cfg.xml` file for all proxy servers in the cell. You will find the `plugin-cfg.xml` in the `<`*`install_root`*`>/<`*`profile_dir`*`>/etc` directory. Select node or server for a smaller 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 > Web servers >** *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 `<`*`install_root`*`>/<`*`profile_dir`*`>/etc/plugin-cfg.xml` directory for the node. 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.

To ensure 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 panel of the WebSphere Application Server administrative console (**Servers** > **Proxy servers** > *server_name* > **HTTP proxy server settings** > **Proxy settings**). This enables the proxy server to honor the WebSphere Application Server private headers that are set by the fronting intermediary server.

Creating a proxy server cluster

This topic describes how to create a cluster of proxy servers, using the **wsadmin** command, that can route requests to applications in a cell.

The cluster includes the machines and nodes that are going to belong to the proxy server cluster. You need to have WebSphere Application Server V6.1 installed. Consider how requests are routed to the proxy server cluster (for example, using domain name server (DNS), Load balancer, or the proxy server). Before you create a cluster, start the deployment manager.

Create a proxy server cluster, as follows:

1. Start the wsadmin utility.
2. Create an empty cluster with no members, by typing:

   ```
   $AdminTask createCluster { -clusterConfig {{<name_of_cluster> true PROXY_SERVER}}}
   ```

   Or, to create a cluster and add a specified proxy server to that cluster, type:

   ```
   $AdminTask createCluster { -clusterConfig {{<name_of_cluster> true PROXY_SERVER}} -convertServer
        {{<node_name> <name_of_proxy_server> "" "" ""}}}
   ```

   This proxy server serves as the template for all subsequent members that are added to the cluster.
3. Add one member at a time to the cluster, as follows:

   ```
   $AdminTask createClusterMember {-clusterName <name_of_cluster -memberConfig
         {-memberNode > <node_name> -memberName <name_of_proxy_server>}}
   ```

   If no members exist in the cluster, the first member that is added serves as the template for subsequent members that are added to the cluster.

   When a proxy server is added to a cluster, proxy-specific configuration settings for it can only be configured using the wsadmin scripting client.
4. Save the configuration changes, as follows:

   ```
   $AdminConfig save
   ```
5. Start the proxy server cluster so that request routing is enabled, as follows:

   ```
   $AdminTask startCluster <name_of_cluster>
   ```
6. 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.

The proxy server cluster is created with the members and is enabled for routing traffic.

Monitor the traffic. See “Monitoring the proxy server with PMI” on page 542 for more information.

Monitoring the proxy server with PMI

You can monitor the traffic of a proxy server using the Performance Monitoring Infrastructure (PMI) function.

The proxy server must be running before performing these steps.

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.

You should know the machines and nodes that will belong to the proxy server cluster before completing these steps. Also, WebSphere Application Server V6.1 needs to be installed on those machines.

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 **wsadmin**.
   b. Get all of the traffic statistics through the proxy server using

   ```
   $AdminControl queryName type=ProxyServer,*
   set proxymbean <cut and paste the MBean identifier from the previous command output>
   $AdminControl getAttribute $proxymbean 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.

The traffic that flows through the proxy server can now be monitored.

Static content caching in the proxy server

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 > Proxy servers >** *<server_name>* **> HTTP proxy server settings > Proxy settings**. You can find trusted security proxies in the **Security** section of this administrative console page.

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 an HTTP 404 response with content from the error page application.

- **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 an HTTP 404 response with content from the error page application.

A sample error application is available in the `<WAS_INSTALL_ROOT>/installableApps/HttpErrorHandler.ear` file.

On Demand Configuration

On Demand Configuration (ODC) enables WebSphere Application Server components, such as the proxy server, to build WebSphere Application Server application deployment, availability, and accessibility information in order to route requests for these applications without any additional configuration at the proxy server.

ODC uses WebSphere Application Server high availability capabilities 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.

ODC is supported on WebSphere Extended Deployment V5.1 and on WebSphere Application Server V6.0 and higher versions of WebSphere Application Server Network Deployment.

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 Application Server proxy 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
	*	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. 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.

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.

Session Initiation Protocol proxy server

The Session Initiation Protocol (SIP) proxy server for WebSphere Application Server is used to initiate 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 back-end SIP resources. The SIP proxy provides a mechanism for other components to extend the base function and support additional deployment scenarios.

The SIP proxy design is based on the WebSphere Application Server 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 WebSphere Application Server 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.

Installing a Session Initiation Protocol proxy server

This topic provides information to install a Session Initiation Protocol (SIP) proxy server.

Ensure that you have a SIP application installed. The SIP container will not listen on a port without a SIP application installed.

Install a SIP proxy server. If you created a cell, deployment manager, and node when you installed WebSphere Application Server, the first five steps do not apply and you can skip to step six.

1. Run the profile creation wizard and create a profile.
2. Run the profile creation wizard and create a deployment manager profile.
3. Start the deployment manager.
4. Federate the node that contains your newly-created profile into the deployment manager cell. You can federate the node in one of the following ways:
   - Type `<`*`WAS_home`*`>profiles/<`*`name_of_profile`*`>/bin/addNode <`*`deployment_manager_host_name`*`> 8879`

     8879 is the default bootstrap port. The server that is being federated should not be running when completing this task from a command line.
   - From the deployment manager administrative console, click **Application Servers > Nodes** and adding the node.

The server in the node that is being federated needs to be running when federating through the deployment manager administrative console.

5. Create a cluster in the administrative console by clicking **Servers** > **Clusters** > **New**. Add the federated server to the cluster.

   Important: Be sure that the default cluster that is defined in the SIP proxy settings points to a valid cluster. Do not use **none**, which is the default.

6. Create a proxy server on the node that you just federated by clicking **Servers** > **Proxy Server** > **New**.

   Important: Be sure to select SIP protocol.

You now have a functional SIP proxy server.

Tip: 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.

Communicating with 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.

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 traverse through the SIP container local domain before being sent on to the remote domain. If a message is received from a remote 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 remote domain because of the trust relationship. An administrator can enable support for external domains in the SIP proxy as follows:

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 remote domains.
2. Ensure that all trusted certificate authorities are set up in the trust store that are 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 are associated with the local domain.
3. Configure all the distinguished names (DNs) that are associated with the remote domains to support. The DN is part of the X.509 certificate that is sent by the remote domain server when the SSL connection is set up. Within the configuration model, each SIP external domain entry includes a field for the remote DN.
4. Assuming that the SIP infrastructure is deployed within each independent domain, provide the DN to the remote domain administrator that is included in local domains public certificate. With this action, the remote domain administrator can configure the proper external domain 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 remote domain. This authorization is based on the agreed upon certificate authorities whose certificates are set up in the local trust store. If the remote 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 remote 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 remote 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.

To trace a SIP proxy server, complete the following steps:

1. Start WebSphere Application Server Network Deployment, 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: `com.ibm.ws.proxy and com.ibm.ws.sip`.
7. Make sure that the **Enable trace with following specification** check box is checked.
8. Click **Apply** → **Save**.

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 in which WebSphere Application Server 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.

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 > Proxy Servers >** *server_name* > **SIP proxy settings**

On the **Configuration** tab, you can edit SIP proxy settings fields.

Default cluster

This settings typically defaults to null and it must be set to a valid cluster before any SIP messages are routed thought 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.

Enable TCP sprayer

Enables and disables SIP outbound request rewriting that enables the SIP proxy to operate behind an IP sprayer.

When this control is checked, TCP host and TCP port are enabled. When this control is unchecked, TCP host and TCP port are disabled.

Data type	Boolean
Default	false

TCP host

This field contains the host name that the SIP proxy writes outbound requests to so that the receiving agent connects back to the IP sprayer.

Enabled only when Enable TCP sprayer is checked.

Data type	String
Default	Blank

TCP port

This field contains the port that the SIP proxy writes outbound requests to so that the receiving agent connects back to the IP sprayer.

Enabled only when enable TCP sprayer is checked.

Range	1 to 65535
Data type	Integer
Default	Blank

Enable SSL sprayer

Enables and disables SIP outbound request rewriting that enables the SIP proxy to operate behind an SSL sprayer.

When this control is checked, SSL host and SSL port are enabled. When this control is unchecked, SSL host and SSL port are disabled.

Data type	Boolean
Default	false

SSL host

This field contains the host name that the SIP proxy writes outbound requests to so that the receiving agent connects back to the SSL sprayer.

Enabled only when Enable SSL sprayer is checked.

Data type	String
Default	Blank

SSL port

This field contains the port that the SIP proxy writes outbound requests to so that the receiving agent connects back to the SSL sprayer.

Enabled only when enable SSL sprayer is checked.

Range	1 to 65535
Data type	Integer
Default	Blank

Enable UDP sprayer

Enables and disables SIP outbound request rewriting that enables the SIP proxy to operate behind a UDP Sprayer.

When this control is checked, UDP host and UDP port are enabled. When this control is unchecked, UDP host and UDP port are disabled.

Data type	Boolean
Default	false

UDP host

This field contains the host name that the SIP proxy writes outbound requests to so that the receiving agent connects back to the UDP sprayer.

Enabled only when Enable UDP sprayer is checked.

Data type	String
Default	Blank

UDP port

This field contains the port that the SIP proxy writes outbound requests to so that the receiving agent connects back to the UDP sprayer.

Enabled only when enable UDP sprayer is checked.

Range	1 to 65535
Data type	Integer
Default	Blank

Enable access logging

This field enables and disables access logging.

When this control is checked, access log maximum size and proxy access log is enabled. When this control is unchecked, access log maximum size and proxy access log is disabled.

Data type	Boolean
Default	Unchecked (false)

Access log maximum size

This field indicates the maximum size, in megabytes, of the access log before it rolls over.

Enabled only when enable access logging is checked.

Range	1 to 65535
Data type	Integer
Default	20

Proxy access log

This field indicates the location of the SIP proxy access log.

Enabled only when enable access logging is checked.

Range	Must be a valid path name
Data type	String
Default	`$(SERVER_LOG_ROOT)/sipproxy.log`

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 > Proxy Servers >** *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 that will be used to make the SIP connection associated with the domain.

Port

The port that is used 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 > Proxy Servers >** *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 SIP traffic to a specific cluster when there is more than one cluster running SIP applications in an 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 > Proxy Servers >** *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 > Proxy Servers >** *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 > Proxy Servers >** *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** > **Proxy Servers** > *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

This panel is used to create or modify a rule. There is one condition type that has a specific set of values, method, while the rest of the condition types are free form text. You can select between fixed method condition selection and free form entry for the other condition types.

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** > **Proxy Servers** > *server_name* > **SIP Proxy Server Settings** > **Routing rules** > *target_cluster* > **Conditions** > **New**.

Condition type

Selects between creating a method condition with a fixed list of values and generic SIP message condition with an arbitrary set of values.

Default	Selected
Setting recommendations	None
Setting dependencies	If this button is selected, the associated drop-down list is enabled. The other entry controls are disabled.

Condition value

Contains the predefined method types used when the SIP method radio button is selected.

Range	INVITE, REGISTER, REFER, SUBSCRIBE, UNSUBSCRIBE, PUBLISH, MESSAGE, OPTIONS, INFO
Default	Invite
Setting recommendations	None
Setting dependencies	None

Condition type: Other

Contains the value against which the SIP message attribute is compared.

Default	Blank
Setting recommendations	None
Setting dependencies	None

SIP proxy inbound channel detail

This panel displays the configuration details of the SIP proxy inbound channel.

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 > Proxy Servers >** *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.

Consult the following list if you are having problems with your proxy server:

- **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 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 routes to another cell.** Review the core group bridge settings. Verify that the peer access point group names match in each cell. Check the peer ports against the bridge interfaces to verify that they are correct. If bridge interfaces or peer ports are added or changed, restart all bridge interfaces.
- **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 536 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 536 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 542 and the custom error page policy section of "Proxy server settings" on page 563.
- **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 536 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 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, for example WebSphere Application Server private headers, of a request, add the originator of the request to the proxy server trusted security proxies list. For example, add an IBM HTTP Server or a plug-in sending requests to the proxy server to the proxy server trusted security proxies list. The plug-in sends WebSphere Application Server private header information that among other things, contains the virtual host information of a request. If the proxy does not trust the WebSphere Application Server private headers from the plug-in (or any client), the proxy server adds its own WebSphere Application Server private headers, which requires the addition of proxy server ports (HTTP and HTTPS) to the virtual host. Most likely, when using the plug-in with the proxy server, the intent is to use the proxy server as a back-end server. Be sure to add the WebSphere Application Server plug-in as a trusted security proxy to avoid having to expose the proxy server ports. Refer to "Routing requests from a plug-in to a proxy server" on page 540 for more information about configuring the WebSphere Application Server plug-in to use with the proxy server. Refer to "Proxy server settings" on page 563 for more information about 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. The problem can be alleviated by setting certain tuning parameters at the operating system level and at the proxy server level that optimize the use of connections for the proxy server:
  - **Operating system tuning for Windows 2000, 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.

– **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

– **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 WebSphere Application Server

Default value	2000
Recommended value	unlimited

– **Operating system tuning for Solaris**

- 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 WebSphere Application Server administrative console by clicking **Servers > Proxy Servers** *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.

You will need to know the machines and nodes that will belong to the proxy server cluster. WebSphere Application Server V6.1 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.

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.

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
   ```

The proxy server and the applications are correctly configured for external access.

Proxy server collection

This topic lists the proxy servers in the cell. A proxy server resides within a WebSphere Application Server Network Deployment 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 > Proxy Servers** .

To configure the proxy server to route work to V6 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 V6 WebSphere Application Server - Express cell requires advanced configuration.

To configure the proxy server to route work to an application server which is not an IBM WebSphere Application Server or a pre-V6 WebSphere Application Server cell, the following advanced configuration is required:

1. Define a generic server cluster. From the administrative console, click **Servers > 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 > Proxy Servers >** *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.

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.

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 > Proxy Servers >** *server_name*

On the **Configuration** tab, you can edit proxy server setting fields.

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** > **Enterprise Applications** > *application_name* page of the administrative console.

Data type	Boolean
Default	true

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 WebSphere Application Server. 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** > **Proxy Servers** > *server_name* > **HTTP Proxy Server Settings** > **Proxy settings**.

You can edit configurable field settings for the proxy server on the Configuration tab.

Content server connection

Configure basic HTTP connection parameters between the proxy server and content servers.

Outbound request timeout: 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: 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: 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: The maximum number of connections that will be pooled to any single content server. Proxy custom properties that tweak content server connections are as follows:

- 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 proxy server name is placed in the host header for content that is not specific to WebSphere Application Server content servers. The options are `true` or `false` and are not case sensitive. The default is false.
- 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.

SSL Configuration: Set the SSL configuration from one of several sources:

Centrally managed	Use the SSL configuration that is scoped for this endpoint.
Specific to this endpoint	Use a specific SSL configuration.
Select SSL Configuration	Options are NONE, CellDefaultSSLSettings, AdminSoapSSLSettings, and NodeDefaultSSLSettings

Caching

The proxy server can be configured to cache the content of servers.

By default, caching content is enabled. The properties that follow apply only if caching is enabled:

- **Enable caching**: Enables caching framework for the proxy server and enables static content caching, as defined by HTTP 1.1 specifications.
- **Cache instance name**: The dynamic cache object cache instance, that is configured in detail under **Resources > Cache instances > Object cache instances**, 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**: Determines 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.

Enable Web services support

Check this option to enable the proxy server to route Web services traffic.

Exclusions

The proxy server examines every incoming request. You can define certain methods for exclusion and if the requested HTTP method matches any of the configured methods for exclusion, the proxy server rejects the requests with a METHOD DISALLOWED error.

Logging

The proxy server has logs that are generated for proxied requests and stored cache requests. With this configuration, you can specify the location of the proxy access log and the cache access log.

Use the default location, or specify a directory location. There is a third log called `${SERVER_LOG_ROOT}/local.log` that logs locally-served proxy content. This content does not come from the proxy cache.

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.

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.

Security

Use this section to set up security options.

- **Trusted security proxies**: Topologies exist where another layer of work routing is enabled on top of the proxy server. For example, Web servers read incoming requests to verify which proxy they are routed to. This configuration field enables intermediaries other than the proxy server to handle the request by explicitly telling the proxy server that is to trust them. Use an IP or fully qualified host name in this field.

  An empty list of trusted security proxies indicates that all WebSphere Application Server plug-in clients are trusted. Once a trusted security proxy is specified, only the listed clients will be trusted.
- **Server header**: Enables configuration of the HTTP server header that is returned to clients. Used to suppress server information. If the value is `""`, the content server name is forwarded to client. If the value is `"TRUE"`, the default server name `"WebSphere Proxy"`, is sent as the content server name. If the value is anything else, the value that is specified is sent as the content server name.

Proxy plugin configuration policy

- **Generate plugin configuration**: Use this parameter for the generation of a proxy plugin configuration file that you can use on a Web server that is deployed in front of the proxy server. The plugin can determine the URI that the proxy is handling on behalf of the application server. The plugin can determine the endpoint, or boundaries of the proxy so that it can properly route requests that it receives to the proxy. This feature is useful for those who prefer to deploy a proven Web server in the demilitarized zone (DMZ), which is fully capable of exploiting the ability of the proxy server.

  Options are available to define a level by which to generate the plugin, as follows:

Scope	Description
None	No scope.
All	The proxy server generates a plugin 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 plugin 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 plugin configuration file only for the proxy server that is currently configured.

- **Plugin config change script**: Specifies the path to a script that is run after the WebSphere Application Server plugin configuration is generated.

Custom error page policy

Use this field to support the use of customized error pages when errors occur during the processing of the request.

The default is no customized error pages generated. The properties that follow enable customized error pages for use when errors occur during request processing:

- **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 remote errors**: When not selected, only HTTP response error status codes generated by the proxy server are handled. When selected, 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. 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**: The status codes that the error page policy provide 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`.

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 WebSphere Application Server or a pre-V6 WebSphere Application Server cell, after creating the proxy server.

To view this administrative console page, click **Servers** > **Generic Server Clusters**.

The system administrator can use the Generic Server Cluster panel to configure external servers, which are non-IBM WebSphere Application Server or a pre-V6 WebSphere Application Server, 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.

Filter

Select the column by which to filter.

Search term(s)

Specifies the filter criteria. This is a user-defined field.

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 WebSphere Application Server or a pre-V6 WebSphere Application Server cell.

To view this administrative console page, click **Servers** > **Generic Server Clusters** > *server_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 > 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.

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 > Generic Server Clusters >** *server_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.

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 > Proxy Servers >** *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 (**Server > 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 WebSphere 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** > **Proxy Servers** > *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 rule

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 last 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).

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.

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 > Proxy servers >** *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

Use this topic to rewrite redirected URLs.

Rewriting rules define how the proxy server will rewrite URLs. The proxy server currently only supports rewriting redirected responses. 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.

To view this administrative console page, click **Servers** > **Proxy servers** > *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`.

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** > **Proxy Servers** > *server_name* > **HTTP Proxy Server Settings** > **Static cache rules**.

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. In order for the rules to have meaning, you must enable caching on the **Servers > Proxy Servers >** *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`.

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* > **Proxy settings> Proxy server transports > HTTP_PROXY_CHAIN > ProxyInboundChannel (PROXY_1)**.

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

Appendix. Directory conventions

References in the product information to *app_server_root*, *profile_root*, and other directories imply specific directory locations. This topic describes the conventions in use for WebSphere Application Server for z/OS.

smpe_root

Refers to the root directory for product code installed with SMP/E.

The corresponding product variable is smpe.install.root.

The default is /usr/lpp/zWebSphere/V6R1.

configuration_root

Refers to the mount point for the configuration file system (formerly, the configuration HFS) in WebSphere Application Server for z/OS.

The configuration_root contains the various app_server_root directories and certain symbolic links associated with them. Each different configuration_root normally requires its own cataloged procedures under z/OS.

The default is /WebSphere/V6R1.

app_server_root

Refers to the top directory for a WebSphere Application Server node.

The node may be of any type—application server, deployment manager, or unmanaged for example. Each node has its own app_server_root. Before Version 6.0 of the product information, this was referred to as the ″WAS_HOME″ directory. Corresponding product variables are was.install.root and WAS_HOME.

The default varies based on node type. Common defaults are *configuration_root*/Appserver and *configuration_root*/DeploymentManager.

profile_root

Refers to the home directory for a particular instantiated WebSphere Application Server profile.

Corresponding product variables are server.root and user.install.root.

In general, this is the same as *app_server_root*/profiles/*profile_name*. On z/OS, this will be always be *app_server_root*/profiles/default because only the profile name ″default″ is used in WebSphere Application Server for z/OS.

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.

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

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Mail Station P300
2455 South Road
Poughkeepsie, NY 12601-5400
USA
Attention: Information Requests

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

Trademarks and service marks

For trademark attribution, visit the IBM Terms of Use Web site (http://www.ibm.com/legal/us/).