IBM Workload Scheduler

Administration

Version 9 Release 4

IBM Workload Scheduler

Administration

Version 9 Release 4

Note
Before using this information and the product it supports, read the information in "Notices" on page 511.

This edition applies to version 9, release 4, modification level 0 of IBM Workload Scheduler (program number 5698-WSH) and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Figures

Tables

About this publication

IBM Workload Scheduler: Administration Guide provides information about the administration of the main components of IBM Workload Scheduler (often called the *engine*).

What is new in this release

Learn what is new in this release.

For information about the new or changed functions in this release, see *IBM Workload Automation: Overview*, section *Summary of enhancements*.

For information about the APARs that this release addresses, see the IBM Workload Scheduler Release Notes at http://www-01.ibm.com/support/docview.wss?rs=672&uid=swg27048863 and the Dynamic Workload Console Release Notes at http://www-01.ibm.com/support/docview.wss?rs=672&uid=swg27048864.

= New or changed content is marked with revision bars. For the PDF format, new or
= changed V9.4 content is marked in the left margin with a pipe (|) character and
= new or changed V9.4FP1 content is marked with an equal sign (=).

Who should read this publication

Learn the audience of this publication.

This publication provides information about the day-to-day administration of the product, and is aimed at the IT administrator or IBM Workload Scheduler IT administrator whose job it is to ensure that the product runs smoothly and correctly. This person will find information about making routine changes to the configuration, for example to add a user, and information about periodic procedures that ensure the integrity of the product, such as backups.

The reader of this book should be an expert systems programmer, who has a reasonable understanding of the IBM Workload Scheduler infrastructure and its inter-component interactions.

Accessibility

Accessibility features help users with a physical disability, such as restricted mobility or limited vision, to use software products successfully.

With this product, you can use assistive technologies to hear and navigate the interface. You can also use the keyboard instead of the mouse to operate all features of the graphical user interface.

For full information, see the Accessibility Appendix in the *IBM Workload Scheduler User's Guide and Reference*.

Technical training

Cloud & Smarter Infrastructure provides technical training.

For Cloud & Smarter Infrastructure technical training information, see:
http://www.ibm.com/software/tivoli/education

Support information

IBM provides several ways for you to obtain support when you encounter a problem.

If you have a problem with your IBM software, you want to resolve it quickly. IBM provides the following ways for you to obtain the support you need:

- Searching knowledge bases: You can search across a large collection of known problems and workarounds, Technotes, and other information.
- Obtaining fixes: You can locate the latest fixes that are already available for your product.
- Contacting IBM Software Support: If you still cannot solve your problem, and you need to work with someone from IBM, you can use a variety of ways to contact IBM Software Support.

For more information about these three ways of resolving problems, see the appendix about support information in *IBM Workload Scheduler: Troubleshooting Guide*.

Chapter 1. Getting started with administration

This publication describes how to perform administrative tasks on IBM Workload Scheduler and Dynamic Workload Console. Many of the procedures described in it require you to identify a file in the installation path of the product and its components. However, these files might be in different installation paths for different components or on different systems, as described in "Where products and components are installed."

Where products and components are installed

This section commences by briefly introducing IBM Workload Automation and includes details about the installation paths and components of IBM Workload Scheduler.

IBM Workload Automation

IBM Workload Automation is the name of a family of products and components, which includes the following:

- IBM Workload Scheduler
- IBM Workload Scheduler for z/OS
- IBM Workload Scheduler for Applications
- Dynamic Workload Console
- IBM Workload Scheduler for Virtualized Data Centres
- IBM Workload Scheduler Plug-in for Informatica PowerCenter

Many IBM Workload Scheduler components are installed in what is called an *IBM Workload Automation instance*.

Installation paths

This section describes the installation paths of the IBM Workload Scheduler components:

TWA_home installation path

As described above, many of the components are installed in an IBM Workload Automation instance. Although this is a notional structure it is represented on the computer where you install IBM Workload Automation components by a common directory referred to in the documentation as TWA_home. The path of this directory is determined when you install a IBM Workload Scheduler component for the first time on a computer. You have the opportunity to choose the path when you make that first-time installation, but if you accept the default path, it is as follows:

Linux `/opt/IBM/TWA<n>`

UNIX `/opt/ibm/TWA<n>`

Windows

```
C:\Program Files\IBM\TWA<n>
```

where <n> is an integer value ranging from <null> for the first instance installed, 1 for the second, and so on.

This path is called, in the publications, *TWA_home*. For details about the directories created outside of *TWA_home*, see *Planning and Installation Guide*.

IBM Workload Scheduler installation path

You can install more than one IBM Workload Scheduler component (master domain manager, backup master domain manager, domain manager, or backup domain manager) on a system, but each is installed in a separate instance of IBM Workload Automation, as described above.

The installation path of IBM Workload Scheduler is:

```
<TWA_home>/TWS
```

IBM Workload Scheduler agent installation path

The agent also uses the same default path structure, but has its own separate installation directory:

```
<TWA_home>/TWS/ITA/cpa
```

Note: The agent also installs some files outside this path. If you have to share, map, or copy the agent files (for example when configuring support for clustering) share, map, or copy these files, as well:

UNIX and Linux operating systems

```
/etc/teb/teb_tws_cpa_agent_<TWS_user>.ini
/opt/IBM/CAP/EMICPA_default.xml
/etc/init.d/tebctl-tws_cpa_agent_<TWS_user>
   (on Linux and Solaris)
/etc/rc.d/init.d/tebctl-tws_cpa_agent_<TWS_user>
   (on AIX)
/sbin/init.d/tebctl-tws_cpa_agent_<TWS_user>
   (on HP-UX)
```

Windows operating systems

```
%windir%\teb\teb_tws_cpa_agent_&lt;tws_user>.ini
%ALLUSERSPROFILE%\Application Data\ibm\CAP\EMICPA_default.xml
```

The agent uses the following configuration files which you might need to modify:

JobManager.ini

This file contains the parameters that tell the agent how to run jobs. You should only change the parameters if advised to do so in the IBM Workload Scheduler documentation or requested to do so by IBM Software Support. Its path is:

```
<TWA_home>/TWS/ITA/cpa/config/JobManager.ini
```

JobManagerGW.ini

When a dynamic agent is installed and **-gateway** `local|remote` is specified, then this file contains the same parameters as the `JobManager.ini` file except for the following differences:

- The **ResourceAdvisorUrl** parameter points to the dynamic workload broker, and not the master domain manager.

The `JobManagerGW.ini` file is installed in the following location:

```
<TWA_home>/TWS/ITA/cpa/config/JobManagerGW.ini
```

ita.ini This file contains parameters which determine how the agent behaves. Changing these parameters may compromise the agent functionality and require it to be reinstalled. You should only change the parameters if advised to do so in the IBM Workload Scheduler documentation or requested to do so by IBM Software Support. Its path is:

<TWA_home>/TWS/ITA/cpa/ita/ita.ini

Installation path for files giving the dynamic scheduling capability
The files that give the dynamic scheduling capability are installed in the following path:

<TWA_home>/TDWB

Dynamic Workload Console installation path
The Dynamic Workload Console can be installed in the path of your choice, but the default installation path is as follows:

On Windows
C:\Program Files\IBM\TWAUI

On UNIX
/opt/IBM/TWAUI

The WebSphere Application Server installation path
The WebSphere Application Server is automatically installed when you create a new *IBM Workload Automation instance*. You can specify any path for the installation. The default installation path is:

<TWA_home>/WAS

For the Dynamic Workload Console: C:\Program Files\IBM\JazzSM

The command line client installation path
The command line client is installed outside all *IBM Workload Automation instances*. Its default path is:

UNIX /opt/ibm/TWS/CLI

Windows
C:\Program Files\IBM\TWS\CLI

The application server tools installation path
Because the WebSphere Application Server is not supplied with an administration GUI, many of its administration tasks are performed by running tools supplied with IBM Workload Scheduler, that perform the required configuration changes. These tools are known as the *wastools*, and are installed in:

<TWA_home>/wastools

However, the information above supplies only the *default* paths. To determine the actual paths of products and components installed in IBM Workload Automation instances, see "Finding out what has been installed in which IBM Workload Automation instances"

Finding out what has been installed in which IBM Workload Automation instances

About this task

If you are not the installer of IBM Workload Scheduler and its components, you might not know what components have been installed, and in which instances of IBM Workload Automation. Follow this procedure to find out:

1. Access the following directory:

UNIX and Linux operating systems
/etc/TWA

Windows operating systems

`%windir%\TWA`

2. List the contents of the directory. Each IBM Workload Automation instance is represented by a file called: `twainstance<instance_number>.TWA.properties`. These files are deleted when all the products or components in an instance are uninstalled, so the number of files present indicates the number of valid instances currently in use.
3. Open a file in a text viewer.

   Attention: Do not edit the contents of this file, unless directed to do so by IBM Software Support. Doing so might invalidate your IBM Workload Scheduler environment.

   The contents are similar to this:

```
TWS_version=9.3.0.0
DB2_basePath=/home/db2inst1/sqllib
DB2_IS_SERVER=TRUE
EWas_basePath=/opt/IBM/WebSphere/AppServer
DB2_INSTANCE_PORT=50000
TWS_counter=1
EWas_counter=1
TWA_path=/opt/tws/tws
TWS_server_name=bvtserver
DB2_ADMINISTRATOR_NAME=db2inst1
TWS_instance_type=MDM
EWas_profile_path=/opt/tws/tws/Appserver/profiles/TWSProfile
EWas_node_name=TWSNode
TWS_basePath=/opt/tws/tws/TWS
EWas_user=tws
EWas_cell_name=TWSCell
EWas_version=8.5.5.4
DB2_version=10.5.0.0
EWas_server_name=server1
EWas_update_installer_dir=
TWS_LAST_COMMITED_LEVEL_KEY=9.3.0.00
TWS_user_name=tws
TWS_FIX_LIST_KEY=
DB2_INSTANCE_NAME=db2inst1
DB2_counter=1
TWA_componentList=TWS,EWas,DB2
EWas_isc_version_key=8.5.5.4
EWas_profile_name=BVTProfile
EWas_service_name=IBMWAS85Service - tws
```

   The important keys to interpret in this file are:

   TWA_path

   This is the base path, to which the installation added one or more of the following directories, depending on what was installed:

   TWS Where the IBM Workload Scheduler component is installed

   TWAUI Where the Dynamic Workload Console is installed

   WAS Where the WebSphere Application Server is installed

   wastools
   Where the tools that you use to configure the WebSphere Application Server are installed

   ssm Where the Netcool® SSM monitoring agent is installed (used in event management)

   TWA_componentList

   Lists the components installed in the instance of IBM Workload Automation

TWS_counter
Indicates if a IBM Workload Scheduler component is installed in this instance of IBM Workload Automation (when the value=1)

TWS_instance_type
Indicates which component of IBM Workload Scheduler is installed in this instance:

MDM Master domain manager

BKM Backup master domain manager

DDM dynamic domain manager

BDDM
Backup dynamic domain manager

FTA Fault-tolerant agent or domain manager

TDWC_counter
Indicates if an instance of Dynamic Workload Console is installed in this instance of IBM Workload Automation (when the value=1)

EWas_counter
Indicates how many applications are installed in this instance of IBM Workload Automation that access the WebSphere Application Server.

TWS_user_name
The ID of the <*TWS_user*> of the IBM Workload Scheduler component.

EWas_user
The ID of the administration user of the WebSphere Application Server. For a default installation, this is the same as the <*TWS_user*>.

The only component of IBM Workload Scheduler which is installed in a IBM Workload Automation instance, but which is not explicitly indicated here, is the Connector. To determine if it has been installed, look at the following combinations of keys:

Agent installed with no Connector
```
TWS_counter=1
EWas_counter=
TWS_instance_type=FTA
TDWC_counter=
TWA_componentList=TWS
```

Agent installed with Connector
```
TWS_counter=1
EWas_counter=1
TWS_instance_type=FTA
TDWC_counter=
TWA_componentList=TWS,EWas
```

Agent installed with no Connector and Dynamic Workload Console
```
TWS_counter=1
EWas_counter=1
TWS_instance_type=FTA
TDWC_counter=1
TWA_componentList=TWS,EWas,TDWC
```

Agent installed with Connector and Dynamic Workload Console
```
TWS_counter=1
EWas_counter=2
TWS_instance_type=FTA
TDWC_counter=1
TWA_componentList=TWS,EWas,TDWC
```

Note: The only difference between these last two is that the `EWas_counter` is 2 instead of 1.

Chapter 2. Customizing and configuring IBM Workload Scheduler

After installing the product you can customize it to fit your operational requirements. You can also change the customized values at any time. This chapter describes the optional customization steps for IBM Workload Scheduler. It is divided into the following sections:

Note: For information about automating the production cycle and managing the production environment, see the *User's Guide and Reference*.

Personalizing UI labels

IBM Workload Scheduler provides the capability to customize user interface labels.

Before you begin

You might find this feature useful for your business users so that the tasks they perform are in the context of your line of business. You can personalize the UI labels for the following UIs:

- Application lab
- Self-Service Catalog and Self-Service Dashboards mobile applications

About this task

The properties file, `whitelabelling.properties`, from which you can modify UI labels must be created manually in a sub-folder named, `Labels`, which you must also create manually in the following path: `<JazzSM_profile_dir>/registry` directory.

Procedure

1. Create a new sub-directory named `Labels` in the following path:

   On Windows:

   ```
   C:\Program Files\IBM\JazzSM\profile\registry
   ```

   On UNIX:

   ```
   /opt/ibm/JazzSM/profile/registry
   ```

2. Create a text file named `whitelabelling.properties` in the sub-directory named `Labels`.

3. Add the following parameters to the `whitelabelling.properties` file and assign a value to the labels you want to modify.

   mobile.title=<*value*>
   ssc.title=<*value*>
   ssd.title=<*value*>
   applab.title=<*value*>
   applab.logo=<*value*>

   where <*value*> corresponds to the following labels:

   Self-Service Catalog and Self-Service Dashboards

   Replace <*value*> with the text to replace the current label:

   - **mobile.title**= <*value*> If defined, this label will appear instead of "IBM Workload Scheduler Mobile Apps"
   - **ssc.title**=<*value*> If defined, this label replaces "Self-Service Catalog"
   - **ssd.title**=<*value*> If defined, this label replaces "Self-Service Dashboards"

   Application Lab

   Replace <*value*> with the text or icon to replace the current values:

   - **applab.title**=<*value*> If defined, this label replaces "Workload Automation" currently found in the browser tab title and in the upper left corner of the Application Lab home page.
   - **applab.logo**=<*value*> If defined, this is the file name of the graphic file that replaces the current IBM logo present in the upper-right corner of the Application Lab UI. This file must be copied to a sub-folder named `logo` in the `Labels` folder and must not exceed 60X30 pixels. For example, to display your company logo in place of the IBM logo, copy the file, `mycompanylogo.gif` in the path: *JazzSM_profile_dir*`>/registry/Labels/logo`.

4. Save your changes.

Setting global options

About this task

Set global options using the **optman** command.

optman

Manages the IBM Workload Scheduler global options. You can list, show and change them.

Authorization

You must have the following security permissions for the global options file in the IBM Workload Scheduler security file to work with this command:

- For `optman ls` or `optman show`:

  ```
  FILE NAME=GLOBALOPTS ACCESS=DISPLAY
  ```

- For `optman chg`:

  ```
  FILE NAME=GLOBALOPTS ACCESS=MODIFY
  ```

See Chapter 4, “Configuring user authorization (Security file),” on page 183 for more information on the security file.

Syntax

optman [-u | -v]
optman [<*connectionParams*>] chg {<*option*> | <*shortName*>} = <*value*>
optman [<*connectionParams*>] ls
optman [<*connectionParams*>] show {<*option*> | <*shortName*>}

Arguments

<connectionParams>
If you are using **optman** from the master domain manager, the connection parameters were configured at installation and do not need to be supplied, unless you do not want to use the default values.

If you are using **optman** from the command line client on another workstation, the connection parameters might be supplied by one or more of these methods:

- Stored in the `localopts` file
- Stored in the `useropts` file
- Supplied to the command in a parameter file
- Supplied to the command as part of the command string

For full details of the connection parameters see "Configuring command-line client access authentication" on page 93.

chg {<*option*> | <*shortName*>} = <*value*>
Change the value of an option to the new value supplied. The option can either be identified by its full or its short name. See "Global options - summary" on page 10 for a table showing all of the options with their full and short names, value ranges and default values. See "Global options - detailed description" on page 16 for a full description of each option.

ls Lists the current values of all global options.

show {<*option*> | <*shortName*>}
Displays the current value of the indicated option. The option can either be identified by its full or its short name. See "Global options - summary" on page 10 for a table showing all of the options with their full and short names, value ranges and default values. See "Global options - detailed description" on page 16 for a full description of each option.

Comments

Some of the changes are effective immediately, but others require a specific action, such as running **`JnextPlan`**, restarting the WebSphere Application Server. These actions are indicated in the option descriptions. See *IBM Workload Scheduler: User's Guide and Reference* for more information on the **`JnextPlan`** command.

Users can decide to maintain an audit trail recording any changes they perform and the related justifications. To enable the justification option, set up in a system shell the IBM Workload Scheduler environment variables listed below before running any **optman** commands:

IWS_DESCRIPTION
Specify the description to be recorded for each change performed by commands in the shell. The maximum length for this value is 512 characters. A warning message displays if you exceed the maximum and excess characters are truncated.

IWS_CATEGORY
Specify the category to be recorded for each change performed by commands in the shell. The maximum length for this value is 128 characters. A warning message displays if you exceed the maximum and excess characters are truncated.

IWS_TICKET
Specify the ticket to be recorded for each change performed by commands in the shell. The maximum length for this value is 128 characters. A warning message displays if you exceed the maximum and excess characters are truncated.

For more information about the justification option, see the section about keeping track of changes in *Dynamic Workload Console User's Guide*.

Examples

Example 1: list the global options
To list all of the global options, when your connection parameters are supplied via the `localopts` and `useropts` files, give the following command:

```
optman ls
```

Example 2: show the value of a global option
To show the current value of the `enCarryForward` global option, identifying it by its short name, give the following command:

```
optman show cf
```

Example 3: change the value of a global option
To change the current value of the `enCarryForward` global option, identifying it by its full name, give the following command:

```
optman chg enCarryForward no
```

Global options - summary

This section summarizes the global options that are managed by **optman**. The columns in the tables have the following meanings:

Description
The brief description of the option

Name The **option** as used in the **optman** commands.

Short name
The **shortName** as used in the **optman** commands.

Default
The default value that is applied to the option at installation (if present).

Range The range or choice of values you can supply (where appropriate).

Units The units that apply to the default and range.

Effect How to make any changes effective. The following codes have been used:

E If you are enabling the option, start the Event Processor. If you are disabling the option, stop the Event Processor.

Imm The change is effective immediately

Imm (DB)
The change is effective immediately in the database only.

J Run **JnextPlan**.

J (Plan)
Run **JnextPlan** - it makes the change effective in the plan only.

NSJ The change is effective on the next submit job stream action.

NSM The change is effective on the next send mail action.

W Restart the WebSphere Application Server.

The following tables summarize the global options for managing the features and functions of IBM Workload Scheduler:

Table 1. Workload service assurance feature

Description	Name	Short name	Default	Range	Units	Effect
Enable workload service assurance	enWorkloadServiceAssurance	wa	yes	yes, no	boolean	J
Approaching late offset	approachingLateOffset	al	120	>=0	seconds	J or W
Deadline offset	deadlineOffset	do	2	>=0	minutes	J or W
Promotion offset	promotionOffset	po	120	>=0	seconds	J
Enable forecast start time calculation	enForecastStartTime	st	no	yes, no	boolean	imm

= *Table 2. Condition-based workflow automation*

Description	Name	Short name	Default	Range	Units	Effect
Name of the job which is automatically added to the plan to run the file monitoring task.	fileStartConditionJobName	fc	file_StartCond	40 bytes		Imm
Name of the job which is automatically added to the plan to resubmit a new instance of the job stream where the start condition is defined.	resubmitJobName	rj	restart_StartCond	40 bytes		Imm
Default offset set for the start condition deadline.	startConditionDeadlineOffset	cd	2400	0001 - 9959	hhmm	Imm

Table 3. Event-driven workload automation feature - general

Description	Name	Short name	Default	Range	Units	Effect
Enable event driven workload automation	enEventDrivenWorkloadAutomation	ed	yes	yes, no	boolean	J or E
Rules deployment frequency	deploymentFrequency	df	5	0-60	minutes	Imm
Enable event processor HTTPS protocol	enEventProcessorHttpsProtocol	eh	yes	yes, no	boolean	J

Table 3. Event-driven workload automation feature - general (continued)

Description	Name	Short name	Default	Range	Units	Effect
IBM event integration facility port for SSL	eventProcessorEIFSslPort	ef	31131	0 - 65535	port number	W and J
IBM event integration facility port	eventProcessorEIFPort	ee	31131	0 - 65535	port number	W and J
EIF Probe server name (used both for events in TEC and TBSM formats)	TECServerName	th	localhost		name	J
EIF Probe server port (used both for events in TEC and TBSM formats)	TECServerPort	tp	5529	0 – 65535	port number	J

Table 4. Event-driven workload automation feature - event mailing

Description	Name	Short name	Default	Range	Units	Effect
Mail sender name	mailSenderName	ms	TWS		name	NSM
SMTP server name	smtpServerName	sn	localhost		name	Imm
SMTP Server port	smtpServerPort	sp	25	0 – 65535	port number	NSM
Mail plug-in uses SMTP authentication	smtpUseAuthentication	ua	no	yes, no	boolean	Imm
SMTP user name	smtpUserName	un	TWS_user		name	Imm
SMTP user password	smtpUserPassword	up				Imm
Mail plug-in uses SSL	smtpUseSSL	us	no	yes, no	boolean	Imm
Mail plug-in uses TLS protocol	smtpUseTLS	tl	no	yes, no	boolean	Imm

Table 5. Event-driven workload automation feature - IBM Workload Scheduler for z/OS plug-in

Description	Name	Short name	Default	Range	Units	Effect
IBM Workload Scheduler for z/OS connector remote server name	zOSRemoteServerName	zr			name	NSJ
IBM Workload Scheduler for z/OS connector server name	zOSServerName	zs	localhost		name	NSJ
IBM Workload Scheduler for z/OS connector server port	zOSServerPort	zp	31217	0 – 65535	port number	NSJ
IBM Workload Scheduler for z/OS connector user name	zOSUserName	zu	TWS_user		name	NSJ
IBM Workload Scheduler for z/OS connector user password	zOSUserPassword	zw				NSJ

Table 6. SSL

Description	Name	Short name	Default	Range	Units	Effect
Enable the SSL full connection	enSSLFullConnection	sf	no	yes, no	boolean	J
Enable strong password encryption	enStrEncrypt	se	no	yes, no	boolean	J

Table 7. Job management

Description	Name	Short name	Default	Range	Units	Effect
Maximum prompts after abend	baseRecPrompt	bp	1000	0 – 65535	prompts	J
Additional prompts after abend	extRecPrompt	xp	1000	0 – 65535	prompts	J
Concurrent access to resources	enExpandedResources	er	yes	yes, no	boolean	J
Automatically grant logon as batch	enLogonBatch	lb	no	yes, no	boolean	J
Long duration job threshold	longDurationThreshold	ld	150	100 - 1000	seconds	J or W
User for binding to remote jobs from shadow job	bindUser	bu	TWS_user			Imm

Table 8. Job stream management

Description	Name	Short name	Default	Range	Units	Effect
Job streams without jobs policy	enEmptySchedsAreSucc	es	no	yes, no	boolean	J
Prevent job stream without "at" dependency from starting	enPreventStart	ps	yes	yes, no	boolean	J

Table 9. Stageman

Description	Name	Short name	Default	Range	Units	Effect
Carry job states	carryStates	cs	null		list of states	J
Enable carry forward	enCarryForward	cf	all	all, no	boolean	J
Enable carry forward for internetwork dependencies	enCFinterNetworkDeps	ci	yes	yes, no	boolean	J
Enable carry forward resource quantity	enCFResourceQuantity	rq	yes	yes, no	boolean	J

Table 9. Stageman (continued)

Description	Name	Short name	Default	Range	Units	Effect
Retain rerun job name	enRetainNameOnRerunFrom	rr	no	yes, no	boolean	J
Remove obsolete job streams	untilDays	ud	0	>=0	days	J

Table 10. Planman

Description	Name	Short name	Default	Range	Units	Effect
Maximum preproduction plan length	maxLen	xl	8	8 - 365	days	J
Minimum preproduction plan length	minLen	ml	8	7 - 365	days	J

Table 11. Logging and auditing

Description	Name	Short name	Default	Range	Units	Effect
Log cleanup frequency	logCleanupFrequency	lc	5	0 - 60	minutes	J
Log history period	logHistory	lh	10	>=0	days	J
Logman minimum and maximum run times policy	logmanMinMaxPolicy	lm	both		literal	J
Logman normal run time calculation policy	logmanSmoothPolicy	lt	-1	0 - 100	factor	J
Enable database auditing	enDbAudit	da	0	0, 1	boolean	Imm
Type of store to be used to log database audit records	auditStore	as	file	db, file, both		Imm
Audit history period	auditHistory	ah	180	>=1	days	Imm

Table 12. Cross dependencies

Description	Name	Short name	Default	Range	Units	Effect
Number of days for retrying to send notifications about job status changes to the remote engine if the notification fails	notificationTimeout	nt	5	1-90	Number	Imm

Table 13. Open Services for Lifecycle Collaboration (OSLC)

Description	Name	Short name	Default	Range	Units	Effect
Description of the IBM Workload Scheduler automation service provider	oslcAutomationDescription	ad			name	Imm
Title of the IBM Workload Scheduler automation service provider	oslcAutomationTitle	at			name	Imm
Host name of the IBM Workload Scheduler service provider (host name of the active master domain manager)	oslcProviderUri	pu			name	Imm
Description of the IBM Workload Scheduler provisioning service provider	oslcProvisioningDescription	pd			name	Imm
Title of the IBM Workload Scheduler provisioning service provider	oslcProvisioningTitle	pt			name	Imm
Password associated with the user who connects to the Registry Services	oslcRegistryPassword	rp			name	Imm
Address of the Registry Services	oslcRegistryUri	cu			name	Imm
User who connects to the Registry Services	oslcRegistryUser	ru			name	Imm

Table 14. SmartCloud Control Desk

Description	Name	Short name	Default	Range	Units	Effect
Address of the SmartCloud Control Desk	sccdUrl	du			name	Imm
User who connects to the SmartCloud Control Desk	sccdUserName	dn			name	Imm
Password associated with the user who connects to the SmartCloud Control Desk	sccdUserPassword	dp			name	Imm

Table 15. ServiceNow

Description	Name	Short name	Default	Range	Units	Effect
Address of the ServiceNow server	servicenowUrl	nu			name	Imm
User who connects to the ServiceNow server	servicenowUserName	nn			name	Imm
Password associated with the user who connects to the ServiceNow server	servicenowUserPassword	np			name	Imm

Table 16. General

Description	Name	Short name	Default	Range	Units	Effect
Company name	companyName	cn			name	J
Enable centralized security in the classic security model	enCentSec	ts	no	yes, no	boolean	J
Enable previous job stream ID	enLegacyId	li	no	yes, no	boolean	J
Evaluate start-of-day	enLegacyStartOfDayEvaluation	le	no	yes, no	boolean	J
Enable list security check	enListSecChk	sc	no	yes, no	boolean	J (Plan) Imm (DB)
Enable plan auditing	enPlanAudit	pa	0	0, 1	boolean	J
Enable security file creation in the role-based security model	enRoleBasedSecurityFileCreation	rs	no	yes,no	boolean	Imm
Enable the fault-tolerant switch manager	enSwfaultTol	sw	no	yes, no	boolean	J
Enable time zones	enTimeZone	tz	yes	yes, no	boolean	J (Plan) Imm (DB)
= Enable What-if Analysis	enWhatIfAnalysis	wi	yes	yes, no	boolean	J
Ignore calendars	ignoreCals	ic	no	yes, no	boolean	J
Start time of processing day	startOfDay	sd	0000	0000 –2359	hhmm	J
Job statistics history period	statsHistory	sh	10	>=0	days	J (Plan) Imm (DB)
/ / Type of accepted license for IBM Workload Scheduler	licenseType	ln	ws	ws, wa , byworkstation		J

Global options - detailed description

This section gives full descriptions of the global options managed by **optman**:

approachingLateOffset | al

Approaching late offset. Used in workload service assurance. The critical start time of a job in the critical network is the latest time that the job can start without causing the critical job to finish after the deadline. In most cases, a job will start well before the critical start time so that if the job runs longer than its estimated duration, the situation does not immediately become critical. Therefore, if a job has not started and the critical start time is only a few minutes away, the timely completion of the critical job is considered to be potentially at risk.

The *approachingLateOffset* option allows you to determine the length of time before the critical start time of a job in the critical network at which you

are to alerted to this potential risk. If a job has still not started the specified number of seconds before the critical start time, the job is added to a hot list that can be viewed on the Dynamic Workload Console.

Note: To qualify for addition to the hot list, all time and follow dependencies must have been resolved.

This option is only active if *enWorkloadServiceAssurance* is set to *yes*.

The default is *120* seconds.

Note: Whatever value you set for this option, if IBM Workload Scheduler loses the connection with its database, the default value is applied to critical job processing, and the warning message AWSJCO135W is issued to tell you what has happened.

Run **JnextPlan** or restart the WebSphere Application Server (**stopappserver** and **startappserver**) to make this change effective.

auditHistory | ah

Audit history period. Used in audit management. This setting applies only when the **auditStore** option is set to `db`. Enter the number of days for which you want to save audit record data. Audit records are discarded on a FIFO (first-in first-out) basis.

The default value is *180* days. This option takes effect immediately.

For more information about auditing, see "Auditing facilities" on page 385.

auditStore | as

Type of store to be used to log database and plan audit records. Enter one of the following:

file To specify that a flat file in the `TWA_home/TWS/audit/database` directory is used to store the audit records. This is the default value.

db To specify that the IBM Workload Scheduler database itself is used to store the audit records.

both To have audit records logged in both the file and the database.

The default value is `both`. Any change of this value is effective immediately.

Note: When you upgrade the master domain manager from a previous release, the default value for this global option is changed. The default value is now **both**. The reason for this change is to support the auditing feature which introduces reporting, versioning and rollback functions for database objects. If you customized the default value in the previous release, the value is overwritten with the new value with the exception of the **auditStore** option with the **DB** value assigned. If the **auditStore** option was set to **DB**, this value is maintained and is not overwritten.

For more information about auditing, see "Auditing facilities" on page 385.

baseRecPrompt | bp

Maximum prompts after abend. Specify the maximum number of prompts that can be displayed to the operator after a job abends.

The default value is *1000*. Run **JnextPlan** to make this change effective.

bindUser | bu

User for binding to remote jobs from shadow job. Specify the user ID that is used to bind a shadow job to a remote job during the security check for "cross dependencies". This user must be given at least the following authorizations in the security file:

- *Display* access to the *job* and *schedule* objects that need to be bound
- *List* access to *job* objects that need to be bound

However, the ID does not need to be in the user registry of the engine, nor have a password, as it is only required for authorization purposes.

The default value is the TWS_user. Any change of this value is effective immediately.

carryStates | cs

Carry job states. A preproduction option that affects the operation of the *stageman* command. Specify the jobs, by state, to be included in job streams that are carried forward. Enclose the job states in parentheses, double quotation marks, or single quotation marks. Commas can be replaced by spaces. The valid internal job states are as follows:

Table 17. Valid internal job states

abend	*abenp*	*add*	*bound*	*done*	*error*	*exec*
fail	*hold*	*intro*	*pend*	*ready*	*rjob*	*sched*
skel	*succ*	*succp*	*suppr*	*susp*	*wait*	*waitd*

Some examples of the option are as follows:

```
carryStates="abend,exec,hold,intro"
carryStates='abend,exec,hold,intro'
carryStates="abend, exec, hold, intro"
carryStates='abend, exec, hold, intro'
```

An empty list is entered as follows:

```
carryStates=null
```

The default value is *null*, which corresponds to selecting all states. Run **JnextPlan** to make this change effective.

companyName | cn

Company name. Specify the name of your company. The maximum length is 40 bytes. If the name contains spaces, enclose the name in double quotation marks ("). If you use the Japanese-Katakana language set, enclose the name within single or double quotation marks.

Run **JnextPlan** to make this change effective.

deadlineOffset | do

Deadline offset. Used in workload service assurance. Used to calculate the critical start of a critical job in the case where a deadline has not been specified neither for the job nor its job stream. In this case the deadline is defaulted to the plan end date and time, plus this offset, expressed in minutes.

This option is only active if *enWorkloadServiceAssurance* is set to *yes*.

The default is *2* minutes.

Note:

1. **Important**: When the plan is extended, the start time of critical jobs with a deadline calculated with this mechanism is automatically changed as a consequence of the fact that it must now match the new plan finishing time.
2. Whatever value you set for this option, if IBM Workload Scheduler loses the connection with its database, the default value is applied to critical job processing, and the warning message AWSJCO135W is issued to tell you what has happened.

Run **JnextPlan** or restart the WebSphere Application Server (**stopappserver** and **startappserver**) to make this change effective.

deploymentFrequency | df

Rules deployment frequency. Used in event rule management. Specify the frequency, in minutes, with which rules are to be checked to detect if there are changes to deploy. All active rules (active rules have the `isDraft` property set to `no` in their definition) that have been changed or added since the last deployment are deployed.

Valid values are in the *0-60* minutes range. If you specify *0*, the changes are not deployed automatically and you must use the **planman deploy** command.

The default value is *5* minutes. The change is effective immediately.

enAddUser | au

Enable the automatic user addition into the Symphony file. This option enables the automatic addition of a user into the Symphony file after you create or modify the user in the database. If you specify "yes", the user is automatically added to the Plan. If you specify "no", the user is not automatically added to the Plan.

The default value is "yes". Changes to this parameter are effective immediately.

For more information about how to use this feature, see "IBM Workload Scheduler: User's Guide and Reference".

enAddWorkstation | aw

Enable the automatic dynamic agent workstation addition into the Symphony file. This option enables the automatic addition of a dynamic agent workstation into the Symphony file after you created the dynamic agent workstation in the database. If you specify "yes", the dynamic agent workstation is automatically added to the Plan. If you specify "no", the dynamic agent workstation is not automatically added to the Plan.

The default is "no". Changes to this parameter are effective immediately.

For more information about how to use this feature, see *IBM Workload Scheduler: User's Guide and Reference*.

enCarryForward | cf

Enable carry forward. A preproduction option that affects the operation of the *stageman* command. Specify if job streams that did not complete are carried forward from the old to the new production plan (Symphony). Enter `yes` to have incompleted job streams carried forward only if the *Carry Forward* option is enabled in the Job Scheduler definition. Enter `all` to have all incompleted job streams carried forward, regardless of the *Carry Forward* option. Enter `no` to completely disable the *Carry Forward* function. If you run the `JnextPlan -for 0000` command and the *Carry Forward* option is set to either `yes` or `no`, a message is displayed informing you that

incompleted job streams might not be carried forward. When the **stageman -carryforward** command is used, it overrides *enCarryForward*. See *IBM Workload Scheduler: User's Guide and Reference* for more information. If this option is set to no, running jobs are moved to the USERJOBS job stream.

The default value is *all*. Run **JnextPlan** to make this change effective.

enCentSec | ts

Enable centralized security. In the classic security model, determine, how the security file is used within the network. Centralized security is not relevant to an end-to-end scheduling environment.

If set to *yes*, the security files of all the workstations of the network can be created and modified only on the master domain manager. In this case, the IBM Workload Scheduler administrator is responsible for their production, maintenance, and distribution.

If set to *no*, the security file of each workstation can be managed by the root user or administrator of the system. The local user can run the *makesec* command to create or update the file.

See *IBM Workload Scheduler: User's Guide and Reference* for more information about centralized security.

The default value is *no*. Run **JnextPlan** to make this change effective.

Note: This option does not apply to role-based security model.

enCFinterNetworkDeps | ci

Enable carry forward for internetwork dependencies. A preproduction option that affects the way **stageman** handles internetwork dependencies. It specifies if external job streams are carried forward from the old to the new production plan (Symphony file). Enter *yes* to have all external job streams carried forward. Enter *no* to have no external job streams carried forward.

The default value is *yes*. Run **JnextPlan** to make this change effective.

enCFResourceQuantity | rq

Enable carry forward resource quantity. A preproduction option that affects the way **stageman** handles resources. Enter *yes* to carry forward the resource quantity from the old production file to the new. Enter *no* to not carry forward the resource quantity. **Stageman** carries forward resource quantities only if the resource is needed by a job or job stream that is also being carried forward. Otherwise the resource quantities are set to the original value. See *IBM Workload Scheduler: User's Guide and Reference* for details on using this feature.

The default value is *yes*. Run **JnextPlan** to make this change effective.

enDbAudit | da

Enable auditing on information available in the database. Enable or disable auditing on information available in the database. To enable auditing on information available in the database, specify *1*. To disable auditing on information available in the database, specify *0*. Auditing information is logged to a flat file in the *TWA_home*/TWS/audit/database directory, to the IBM Workload Scheduler database itself, or to both. To choose which, set the **optman** property *auditStore*. Each IBM Workload Scheduler workstation maintains its own log. Only actions are logged, not the success or failure of the action. Installation of dynamic domain managers and agents is not recorded in audit logs.

The default value is *1*. Changes to this parameter are effective immediately.

Note: When you upgrade the master domain manager from a previous release, the default value for this global option is changed. The default value is now **1**. The reason for this change is to support the auditing feature which introduces reporting, versioning and rollback functions for database objects. If you customized the default value in the previous release, the value is overwritten with the new value.

For more information about auditing, see "Auditing facilities" on page 385.

enEmptySchedsAreSucc | es
Job streams without jobs policy. Specify the behavior of job streams without any jobs. If set to *yes*, the job streams that contain no jobs are set to SUCC after their dependencies are resolved. If set to *no*, the job streams are left in READY status.

The default value is *no*. Run **JnextPlan** to make this change effective.

enEventDrivenWorkloadAutomation | ed
Enable event-driven workload automation. Enable or disable the event-driven workload automation feature. To enable, specify *yes*. To disable, specify *no*.

The default value is *yes*.

After disabling, you must run **JnextPlan** and stop the event processing server (with the **conman stopevtp** command).

After enabling, you must run **JnextPlan** and start the event processing server (with the **conman startevtp** command).

enEventDrivenWorkloadAutomationProxy | pr
Enable event-driven workload automation proxy. Enable or disable the event-driven workload automation proxy feature. To enable, specify *yes*. To disable, specify *no*.

The default value is *no*. Run **JnextPlan** to make this change effective.

enEventProcessorHttpsProtocol | eh
Enable event processor HTTPS protocol. Used in event rule management. Enables or disables the use of the HTTPS protocol to connect to the event processor server. To enable, enter *yes*. To disable, enter *no*.

The default value is *yes*. Run **JnextPlan** to make this change effective.

enExpandedResources
Enables up to 60 concurrent holders for an IBM Workload Scheduler resource. Enter `yes` to enable up to 60 concurrent holders for a resource. Enter `no` to disable the feature and use only 32 holders for a resource.

The default value is `yes`. Run JnextPlan to make this change effective.

enForecastStartTime | st
Enable forecast start time. Only applicable when workload service assurance is enabled (see *enWorkloadServiceAssurance*). Enter *yes* to enable the calculation of the predicted start time of each job when running a forecast plan: this option is recommended if you want to take advantage of the enhanced forecast capability that calculates the start time of each job considering the estimated duration of its predecessor jobs. Enabling this feature could negatively impact the time taken to generate the forecast plan. Enter *no* to disable the calculation of the predicted start time of each job when running a forecast plan.

The default value is *no*. Any change of this value is effective immediately.

When this option is set to *yes*, the **enPreventStart** global option is ignored during the creation of forecast plans.

enLegacyId | li

Enable previous job stream ID. Determine how job streams are to be named when operating in mixed environments with versions of IBM Workload Scheduler older than version 8.3, managed by a version 8.5 master domain manager. This option is not supported by the Self Service catalog, which ignores it even if its value is set to YES.Use this option to keep consistency in identifying the job streams in the plan. The value assigned to this option is read either when the production plan is created or extended, or when submitting job streams in production using conman.

When the plan is created or extended, if this option is set to *no*, the Job Scheduler instance is assigned a new ID following the normal mechanism of IBM Workload Scheduler. In the Symphony file, the Job Scheduler name is equal to this ID. If the option is set to *yes*, the Job Scheduler instance is assigned an ID (symphony ID) equal to the Job Scheduler name. In the Symphony file the Job Scheduler name is equal to the real Job Scheduler name. If more instances of the same Job Scheduler are present, an ID is generated for every instance, with an alias that starts with the Job Scheduler name.

The default value is *no*. Run **JnextPlan** to make this change effective.

enLegacyStartOfDayEvaluation | le

Evaluate start-of-day. Specify how the *startOfDay* option is to be managed across the IBM Workload Scheduler network. If you set this option to *yes*, the *startOfDay* value on the master domain manager is converted to the local time zone set on each workstation across the network. If you set this option to *no*, the *startOfDay* value on the master domain manager is applied as is on each workstation across the network. This option requires that the *enTimeZone* option is set to *yes* to become operational.

The default value is *no*. Run **JnextPlan** to make this change effective.

enListSecChk | sc

Enable list security check. Control the objects in the plan that a user is permitted to list when running a query on the Dynamic Workload Console or a **conman show <object>** command. If set to *yes*, objects in the plan returned from a query or show command are shown to the user only if the user has been granted the list permission in the security file. If set to *no*, all objects are shown, regardless of the settings in the security file.

Note: Setting this option to *yes* affects how the graphical user interfaces function for the users defined in the security file.

The default value is *no*. Run **JnextPlan** to make this change effective for the plan. For the database, this option takes immediate effect.

enLogonBatch | lb

Automatically grant logon as batch. This is for Windows jobs only. If set to *yes*, the logon users for Windows jobs are automatically granted the right to *Logon as batch job*. If set to *no*, or omitted, the right must be granted manually to each user or group. The right cannot be granted automatically for users running jobs on a backup domain manager, so you must grant those rights manually.

The default value is *no*. Run **JnextPlan** to make this change effective.

enPlanAudit | pa

Enable plan auditing. Enable or disable auditing on information available in the plan. To enable auditing on information available in the plan, specify *1*. To disable auditing on information available in the plan, specify *0*. Auditing information is logged to a flat file in the *TWA_home*/TWS/audit/plan directory. Auditing information is logged to a flat file in the *TWA_home*/TWS/audit/database directory, to the IBM Workload Scheduler database itself, or to both. Each IBM Workload Scheduler workstation maintains its own log. For the plan, only actions are logged in the auditing file, not the success or failure of any action.

For more information about auditing, see "Auditing facilities" on page 385.

The default value is *1*. Changes to this parameter are effective immediately.

Note: When you upgrade the master domain manager from a previous release, the default value for this global option is changed. The default value is now **1**. The reason for this change is to support the auditing feature which introduces reporting, versioning and rollback functions for database objects. If you customized the default value in the previous release, the value is overwritten with the new value.

enPreventStart | ps

Prevent job stream without "at" dependency from starting. Specify if job streams without an *at* dependency are to be prevented from starting immediately, without waiting for the run cycle specified in the Job Scheduler. Valid values are *yes* and *no*.

The default value is *yes*. Run **JnextPlan** to make this change effective.

When the **enForecastStartTime** option is set to *yes*, this option is ignored during the creation of forecast plans.

enRetainNameOnRerunFrom | rr

Retain rerun job name. A production option that affects the operation of **Batchman**, the production control process of IBM Workload Scheduler. Its setting determines if jobs that are rerun with the **Conman** *rerun* command retain their original job names. To have rerun jobs retain their original job names, enter *yes*. Enter *no* to assign the *rerun from* name to rerun jobs.

The default value is *no*. Run **JnextPlan** to make this change effective.

enRoleBasedSecurityFileCreation | rs

Enable the role-based security model. This option enables the automatic creation of the security file using the role-based security model. You define the role-based security model in the master domain manager database by using the **Manage Workload Security** interface from Dynamic Workload Console or the **composer** command-line program.

The default value is *no*, which means that the role-based security model is not enabled for your installation. You continue to use the classic security model that allows you to update your security file by using **dumpsec** and **makesec** commands from the command line.

At any time, specify *yes* if you want to enable the role-based security model and replace your current security file. A new security file is created and updated with the security objects (domains, roles, and access control lists) that you define in the master domain manager database by using the **Manage Workload Security** interface from Dynamic Workload Console.

For more information about how to use this option, see Chapter 4, "Configuring user authorization (Security file)," on page 183.

Changes to this parameter are effective immediately.

enSSLFullConnection | sf
Enable the SSL full connection. Specify that IBM Workload Scheduler uses a higher level of SSL connection than the standard level. For full details see "Configuring full SSL security" on page 332. Valid values are *yes* to enable the SSL full connection or *no* to disable the SSL full connection.

The default value is *no*. Run **JnextPlan** to make this change effective.

enStrEncrypt | se
Enable strong password encryption. Enable or disable strong encryption. Enable strong encryption by setting this option to *yes*. See "Configuring the SSL connection protocol for the network" on page 332.

The default value is *no*. Run **JnextPlan** to make this change effective.

enSwfaultTol | sw
Enable the fault-tolerant switch manager. Enable or disable the fault-tolerant switch manager feature. Valid values are *yes* to enable the fault tolerant switch manager, and *no* to disable it. This option has not dynamic capabilities and is not designed to work with broker agents. It applies to fault-tolerant agents. See the *IBM Workload Scheduler: User's Guide and Reference* for more details.

The default value is *no*. Run **JnextPlan** to make this change effective.

enTimeZone | tz
Enable time zones. Enables or disables the time zone option. To activate time zones in your network, specify *yes*. To disable time zones in the network, specify *no*. See "Enabling the time zone feature" on page 98.

The default value is *yes*. Run **JnextPlan** to make this change effective in the plan. For the database, this option takes effect immediately.

= **enWhatIfAnalysis | wi**
= **Enable What-if Analysis.** Enables or disables What-if Analysis, which is
= the feature that shows plan activities displayed against time and give you
= a visual representation of your plan at a glance in real time. To enable
= What-if Analysis, specify *yes*. To disable What-if Analysis, specify *no*. See
= *Dynamic Workload Console User's Guide* for details on using this feature..

= The default value is *yes*. Run **JnextPlan** to make this change effective.

enWorkloadServiceAssurance | wa
Enable workload service assurance. Enables or disables workload service assurance, which is the feature that manages the privileged processing of mission critical jobs and their predecessors. Specify *yes* to enable or *no* to disable.

Note: Before starting to use workload service assurance you must set up the *TWS_user* in the security file to have the appropriate access to the objects that this feature will modify - see "The *TWS_user* - special security file considerations" on page 236

The default value is *yes*. Run **JnextPlan** to make this change effective.

eventProcessorEIFSslPort | ef
Tivoli® event integration facility port. Used in event rule management.

Specify the port number for SSL where the event processor server receives events from the Tivoli Event Integration Facility (EIF). Valid values are in the *0-65535* range.

The default value is *31131*. If you change the value, restart the WebSphere Application Server (**stopappserver** and **startappserver**) and run **JnextPlan** to make this change effective.

eventProcessorEIFPort | ee
Tivoli event integration facility port. Used in event rule management. Specify the port number where the event processor server receives events from the Tivoli Event Integration Facility (EIF). Valid values are in the *0-65535* range.

The default value is *31131*. If you change the value, restart the WebSphere Application Server (**stopappserver** and **startappserver**) and run **JnextPlan** to make this change effective.

If you use a security firewall, make sure this port is open for incoming and outgoing connections.

extRecPrompt | xp
Additional prompts after abend. Specify an additional number of prompts for the value defined in *baseRecPropmt*. This applies when a job is rerun after abending and the limit specified in *baseRecPropmt* has been reached.

The default value is *1000*. Run **JnextPlan** to make this change effective.

= **fileStartConditionJobName | fc**
= **Name of the job in charge of running the file monitoring task .**
= Applicable only if you select file as the start condition type. The name of
= the job which is automatically added to the plan to run the file monitoring
= task. This value is used by default if you do not specify any value for the
= job name when defining the start condition. If you specify a value for the
= job name, this value is ignored.

= The default value is *file_StartCond*. The maximum length is 40 bytes.
= Changes to this parameter are effective immediately.

ignoreCals | ic
Ignore calendars. A preproduction option that affects the operation of the **planman** command. Its setting determines if user calendars are copied into the new production plan (Symphony) file. To prevent user calendars from being copied into the new production plan, enter *yes*.

The default value is *no*. See *IBM Workload Scheduler: User's Guide and Reference*. Run **JnextPlan** to make this change effective.

/ **licenseType | ln**
/ **Type of accepted license for IBM Workload Scheduler.**

/ Supported values are:

/ **ws**
/ **perServer**
/ to specify the IBM Workload Scheduler Processor Value Unit (PVU)
/ pricing.

/ **wa**
/ **perJob**
/ to specify the IBM Workload Scheduler Per Job (PJ) pricing.

/ **byWorkstation**
/ to specify that the licensing type (either **perServer** or **perJob**) is
/ specified at creation time for each workstation. For more
/ information about defining workstations, see the section about
/ workstation definition in *User's Guide and Reference*.

/ The default value is **perServer**. Run **JnextPlan** to make this change
/ effective. For additional information about license management and
/ metrics, see Chapter 13, "License Management in IBM License Metric Tool," on page 501. You can define this option for the following
/ workstation types:

/ • master domain manager
/ • fault-tolerant agent
/ • standard agent
/ • dynamic agent

logCleanupFrequency | lc
Log cleanup frequency. Used in event rule and audit management . Specify how often the automatic cleanup of log instances is run. Valid values are in the *0-60* minutes range. If you specify *0*, the automatic cleanup feature is disabled.

The default value is *5* minutes. This option takes effect immediately.

logHistory | lh
Log history period. Used in event rule management. Enter the number of days for which you want to save rule instance, action run, and message log data. Log instances are discarded on a FIFO (first-in first-out) basis.

The default value is *10* days. This option takes effect immediately.

logmanMinMaxPolicy | lm
Logman minimum and maximum run times policy. Specify how the minimum and maximum job run times are logged and reported by **logman**. Possible values are:

elapsedtime
The minimum and maximum elapsed runtimes are logged and reported.

cputime
The minimum and maximum CPU run times are logged and reported.

both Both the minimum and maximum job runtimes are logged and reported.

See *IBM Workload Scheduler: User's Guide and Reference* for details on using this feature.

The default value is *both*. Run **JnextPlan** to make this change effective.

logmanSmoothPolicy | lt
Logman normal run time calculation policy. Set the weighting factor that favors the most recent job run when calculating the normal (average) run time for a job. This is expressed as a percentage. For example, specify *40* to apply a weighting factor of 40% to the most recent job run, and 60% to the existing average. See *IBM Workload Scheduler: User's Guide and Reference* for more information about how to use this option.

The default value is *10*. Run **JnextPlan** to make this change effective.

longDurationThreshold | ld
Long duration job threshold. Specify, when comparing the actual duration of a job to the estimated duration, the threshold over which the job is considered to be of "long duration." The threshold value is expressed as a percentage with respect to the estimated duration. For example, if the threshold is set to *150*, and the actual duration is more than 150% of the estimated duration (it is 50% greater), the job is considered to be a "long duration" job.

If you have the workload service assurance feature enabled, the effect of a "critical" job satisfying the long duration criteria is that the job is inserted automatically into the hot list.

Valid values are between:
100 The minimum value. All jobs that exceed the estimated duration are considered long duration jobs
1000 The maximum value. Only those jobs that last ten times as long as their estimated duration are considered as long duration jobs

The default is *150*.

Note: Whatever value you set for this option, if you have the workload service assurance feature enabled, and IBM Workload Scheduler loses the connection with its database, the default value is applied to critical job processing, and the warning message AWSJCO135W is issued to tell you what has happened.

Run **JnextPlan** or restart the WebSphere Application Server (**stopappserver** and **startappserver**) to make this change effective.

mailSenderName | ms
Mail sender name. Used in event rule management. If you deploy rules implementing an action that sends emails via an SMTP server, specify a string to be used as the sender of the emails.

The default value is *TWS*. Changes to this parameter are effective for the next mail send action performed.

maxLen | xl
Maximum preproduction plan length. Specify the maximum length of the preproduction plan in days after it is automatically extended or created. The value for *maxLen* must be greater than or equal to the value for *minLen* and must be in the range of *8* to *365*.

The default is *14* days. Run **JnextPlan** to make this change effective.

minLen | ml
Minimum preproduction plan length. Specify the minimum length in days of the preproduction plan that can pass after the production plan is created or extended, without extending the preproduction plan. If the days left in the preproduction plan after a **JnextPlan** are less than the value of this option, the preproduction plan is automatically extended. The value for *minLen* must be less than or equal to the value for *maxLen* and must be in the range of *7* to *365*.

The default is *8* days. Run **JnextPlan** to make this change effective.

notificationTimeout | nt
Notification timeout. Used in cross dependencies. Specify how many days IBM Workload Scheduler must retry sending notifications about job status

changes to the remote engine if the notification fails. When this timeout expires, the job request subscription and the status notifications associated to this job are discarded.

Valid values are in the range of *1* to *90*. The default is *5* days. Changes to this parameter are effective immediately.

oslcAutomationDescription | ad
Description of the IBM Workload Scheduler automation service provider. Used in OSLC integration to register the IBM Workload Scheduler automation service provider in the Registry Services. This value is used to define a description for the service provider.

Changes to this parameter are effective immediately.

oslcAutomationTitle | at
Title of the IBM Workload Scheduler automation service provider. Used in OSLC integration to register the IBM Workload Scheduler automation service provider in the Registry Services. This value is used to uniquely identify the automation service provider. To easily identify the service provider you want to use, use a meaningful title for each IBM Workload Scheduler automation service provider registered in the same Registry Services.

Changes to this parameter are effective immediately.

oslcProviderUri | pu
Address of the IBM Workload Scheduler service provider. Used in OSLC integration to register the IBM Workload Scheduler service provider in the Registry Services. Use the format `https://hostname:port`, where `hostname` is the name of the host used to connect to the master domain manager. For example, `https://myProviderHostanme.com:31115`.

Changes to this parameter are effective immediately.

oslcProvisioningDescription | pd
Description of the IBM Workload Scheduler automation service provider. Used in OSLC integration to register the IBM Workload Scheduler automation service provider in the Registry Services. This value is used to define a description for the service provider.

Changes to this parameter are effective immediately.

oslcProvisioningTitle | pt
Title of the IBM Workload Scheduler provisioning service provider. Used in OSLC integration to register the IBM Workload Scheduler provisioning service provider in the Registry Services. This value is used to uniquely identify the provisioning service provider. To easily identify the service provider you want to use, use a meaningful title for each IBM Workload Scheduler provisioning service provider registered in the same Registry Services.

Changes to this parameter are effective immediately.

oslcRegistryPassword | rp
Password of the user connecting to the Registry Services. Used in OSLC integration to register the IBM Workload Scheduler service provider in the Registry Services.

Changes to this parameter are effective immediately.

oslcRegistryUri | cu
Address of the Registry Services. Used in OSLC integration to register the

IBM Workload Scheduler service provider in the Registry Services. Use the format `https://hostname:port/oslc/pr`.

Changes to this parameter are effective immediately.

oslcRegistryUser | ru
User connecting to the Registry Services. Used in OSLC integration to register the IBM Workload Scheduler service provider in the Registry Services.

Changes to this parameter are effective immediately.

promotionOffset | po
Promotion offset. Used in workload service assurance. Specify when a job become eligible for promotion in terms of the number of seconds before its critical start time is reached. Applies only to jobs that are flagged as critical in a job stream definition and to their predecessor jobs. A critical job and its predecessors make up a critical network.

When a predecessor jeopardizes the timely completion of the critical job, it is *promoted*; that is, it is assigned additional resources and its submission is prioritized with respect to other jobs that are out of the critical network. Also critical jobs might be promoted.

The scheduler calculates the critical start time of a critical job by subtracting its estimated duration from its deadline. It calculates the critical start time of a critical predecessor by subtracting its estimated duration from the critical start time of its next successor. Within a critical network the scheduler calculates the critical start time of the critical job first and then works backwards along the chain of predecessors. These calculations are reiterated as many times as necessary until the critical job has run.

This option is only active if *enWorkloadServiceAssurance* is set to *yes*.

The default is *120* seconds.

Run **JnextPlan** to make this change effective.

= **resubmitJobName | rj**
= **Name of the job in charge of resubmitting the job stream.** Specify the
= name of the Job Stream Submission job which is automatically added to
= the plan to resubmit a new instance of the job stream where the start
= condition is defined.

= The default value is *MASTERAGENTS#restart_StartCond*, where
= MASTERAGENTS is the name of the pool workstation on which the Job
= Stream Submission job runs. The maximum length for the workstation
= name is 16 bytes, and the maximum length for the job name is 40 bytes.
= Changes to this parameter are effective immediately.

/ **resubmitJobUserName | rw**
/ **Name of the user in charge of resubmitting the job stream.** Specify the
/ user name which owns the Job Stream Submission job. The Job Stream
/ Submission job is automatically added to the plan to resubmit a new
/ instance of the job stream where the start condition is defined.

/ The default value is *TWS_User*. Changes to this parameter are effective
/ immediately. If the user defined in the **resubmitJobUserName** property
/ does not exist, the user name and password defined on the WebSphere
/ Application Server installed on the master domain manager or backup
/ master domain manager are used. This implies that the user defined in the
/ **resubmitJobUserName** property must be the same both on the master

/ domain manager and on the backup master domain manager, or must be
/ changed immediately after switching the master domain manager.

sccdUrl | du
IBM SmartCloud Control Desk URL. Used in event rule management. If you use rules that implement an action that opens a ticket to an IBM SmartCloud Control Desk server (or any other application that can open ticket in IBM SmartCloud Control Desk format), specify the IBM SmartCloud Control Desk URL. You can change this value when you define the action if you want to use a different IBM SmartCloud Control Desk URL.

The default value is "`http://localhost:8080/maximo/oslc/os/oslcincident`". Changes to this parameter are effective immediately.

sccdUserName | dn
SmartCloud Control Desk user name. Used in event rule management. If you deploy rules that implement an action that opens a ticket by using the SmartCloud Control Desk, specify the identifier of the user connecting to the SmartCloud Control Desk server.

The default value is the IBM Workload Scheduler user on the master domain manager. Changes to this parameter are effective immediately.

sccdUserPassword | dp
SmartCloud Control Desk user password. Used in event rule management. If you deploy rules that implement an action that opens a ticket by using the SmartCloud Control Desk, specify the password associated with the user connecting to the SmartCloud Control Desk server. The password is stored in an encrypted form.

Changes to this parameter are effective immediately.

| **servicenowUrl | nu**
| **ServiceNow URL.** Used in event rule management. If you use rules that implement an action that opens an incident in ServiceNow (or any other application that can open an incident in the ServiceNow format), specify the ServiceNow URL. You can change this value when you define the action if you want to use a different ServiceNow URL.

| The default value is "`http://localhost:8080/api/now/table/incident`". Changes to this parameter are effective immediately.

| **servicenowUserName | nn**
| **ServiceNow user name.** Used in event rule management. If you use rules that implement an action that opens an incident in ServiceNow, specify the identifier of the user connecting to the ServiceNow server.

| The default value is the IBM Workload Scheduler user on the master domain manager. Changes to this parameter are effective immediately.

| **servicenowUserPassword| np**
| **ServiceNow user password.** Used in event rule management. If you use rules that implement an action that opens an incident in ServiceNow, specify the password associated with the user connecting to the ServiceNow server.

| Changes to this parameter are effective immediately.

smtpServerName | sn
SMTP server name. Used in event rule management. If you deploy rules

implementing an action that sends emails via an SMTP server, specify the name of the SMTP server to be used by the mail plug-in.

The default value is *localhost*. Changes to this parameter are effective immediately.

smtpServerPort | sp
SMTP Server port. Used in event rule management. If you deploy rules implementing an action that sends emails via an SMTP server, specify the port number used to connect to the SMTP server by the mail plug-in. Valid values are in the range *0–65535*.

The default value is *25*. Changes to this parameter are effective for the next mail send action performed.

smtpUseAuthentication | ua
Mail plug-in uses SMTP authentication. Used in event rule management. If you deploy rules implementing an action that sends emails via an SMTP server, specify if the SMTP connection needs to be authenticated. Values are *yes* or *no*.

The default is *no*. Changes to this parameter are effective immediately.

smtpUserName | un
SMTP server user name. Used in event rule management. If you deploy rules implementing an action that sends emails via an SMTP server, specify the SMTP server user name.

The default value is the name of the IBM Workload Scheduler user (the TWS_user) on the master domain manager. Changes to this parameter are effective immediately.

smtpUserPassword | up
SMTP server user password. Used in event rule management. If you deploy rules implementing an action that sends emails via an SMTP server, specify the SMTP server user password. The password is stored in an encrypted form.

Changes to this parameter are effective immediately.

smtpUseSSL | us
Mail plug-in uses SSL. Used in event rule management. If you deploy rules implementing an action that sends emails via an SMTP server, specify if the SMTP connection is to be authenticated via SSL. Values are *yes* or *no*.

The default is *no*. Changes to this parameter are effective immediately.

smtpUseTLS | tl
Mail plug-in uses TLS protocol. Used in event rule management. If you deploy rules implementing an action that sends emails via an SMTP server, specify if the SMTP connection is to be authenticated via the Transport Layer Security (TLS) protocol. Values are *yes* or *no*.

The default is *no*. Changes to this parameter are effective immediately.

startOfDay | sd
Start time of processing day. Specify the start time of the IBM Workload Scheduler processing day in 24-hour format: *hhmm* (*0000-2359*).

The default value is *0000* (0:00 a.m.), but if you upgraded your environment to version 9.4 starting from a version earlier than version 8.6, the default value is *0600* (6:00 a.m.). If you change this option, you must

also change the launch time of the *final* Job Scheduler, which is usually set to one minute before the start time. Run **JnextPlan** to make the change of *startOfDay* effective.

statsHistory | sh
Job statistics history period. Specify the number of days for which you want to maintain job statistics. Statistics are discarded on a FIFO (first-in first-out) basis. For example, if you leave the default value of *400*, statistics are maintained for the last 400 days. This has no effect on job standard list files, which must be removed with the *rmstdlist* command. See the *IBM Workload Scheduler: User's Guide and Reference* for information about the *rmstdlist* command.

The default value is *400*. Run **JnextPlan** to make this change effective in the plan. For the database, this option takes effect immediately.

= **startConditionDeadlineOffset | cd**
= **Start condition deadline offset.** The default offset set for the start
= condition deadline in 24 hour format: "hhmm" (0001-9959). Specify the time
= range during which the start condition is active.

= The default value is *2400* and the range is *0001 - 9959*. Changes to this
= parameter are effective immediately.

TECServerName | th
EIF Probe server name. Used in event rule management. If you use rules implementing an action that forwards events to a Tivoli Enterprise Console® or Tivoli Business Service Manager server (or any other application that processes events in TEC or TBSM format), specify the EIF Probe server name. If you want to use a different EIF Probe server, you can change this value when you define the action.

The default is *localhost*. Run **JnextPlan** to make this change effective.

TECServerPort | tp
EIF Probe server port. Used in event rule management. If you use rules implementing an action that forwards events to a Tivoli Enterprise Console or Tivoli Business Service Manager server (or any other application that processes events in TEC or TBSM format), specify the port number of the EIF Probe server. If you want to use a different EIF Probe server, you can change this value when you define the action.

The default port number is *5529*. Run **JnextPlan** to make this change effective.

| **untilDays | ud**
| **Remove obsolete job stream instances from the plan.** If an **until** time
| (latest start time) has not been specified for a job stream, then the default
| **until** time is calculated adding the value of this option, expressed in
| number of days, to the scheduled time for the job stream. If the
| *enCarryForward* option is set to **all**, and the number of days specified for
| *untilDays* is reached, then any job stream instances in the plan that ended
| in error are automatically removed from the plan and not added to the
| new production plan.

| The default value is **0**. If the default value is used, then no default time is
| set for the **until** time (latest start time).

| Run **JnextPlan** to make this change effective.

workstationLimit | wl
The workstation limit.

Used in the automatic dynamic agent registration. This parameter specifies the dynamic agent workstation limit value that the dynamic agent workstation assumes after the workstation is added to the plan. You can later modify the dynamic agent workstation limit value by using the **conman** command line or the Dynamic Workload Console.

Valid values are in the *0-1024* range.

The default is *100*. Changes to this parameter are effective immediately.

zOSRemoteServerName | zr

IBM Workload Scheduler for z/OS connector remote server name. Used in event rule management. If you deploy rules implementing an action that submits job streams to the IBM Workload Scheduler for z/OS controller, enter the name of the controller specified as the engine to the z/OS connector. It must exactly match the z/OS connector engine name and is case sensitive.

After changing the value of this parameter, the change becomes effective when the next `submit` action is run.

zOSServerName | zs

IBM Workload Scheduler for z/OS connector server name. Used in event rule management. If you deploy rules implementing an action that submits job streams to the IBM Workload Scheduler for z/OS controller, specify the name or the hostname of the system where the IBM Workload Scheduler for z/OS connector runs. The default value is `localhost`.

After changing the value of this parameter, the change becomes effective when the next `submit` action is run.

zOSServerPort | zp

IBM Workload Scheduler for z/OS connector server port. Used in event rule management. If you deploy rules implementing an action that submits job streams to the IBM Workload Scheduler for z/OS controller, specify the bootstrap port number of the IBM Workload Scheduler for z/OS connector server. Valid values are in the range 0-65535. The default value is 31217.

After changing the value of this parameter, the change becomes effective when the next `submit` action is run.

zOSUserName | zu

IBM Workload Scheduler for z/OS connector user name. Used in event rule management. If you deploy rules implementing an action that submits job streams to the IBM Workload Scheduler for z/OS controller, specify the IBM Workload Scheduler for z/OS connector user name required to access the IBM Workload Scheduler for z/OS engine.

After changing the value of this parameter, the change becomes effective when the next `submit` action is run.

zOSUserPassword | zw

IBM Workload Scheduler for z/OS connector user password. Used in event rule management. If you deploy rules implementing an action that submits job streams to the IBM Workload Scheduler for z/OS controller, specify the IBM Workload Scheduler for z/OS connector user password required to access the IBM Workload Scheduler for z/OS engine. The password is stored in encrypted form.

After changing the value of this parameter, the change becomes effective when the next `submit` action is run.

Setting local options

Set local options, such as general attributes of the workstation for the IBM Workload Scheduler processes, in the `localopts` file. Changes do not take effect until **netman** is stopped (**conman shut;wait**) and restarted (**StartUp**).

During the installation process, a working copy of the local options file is installed as *TWA_home*/TWS/localopts.

The `localopts` file is not modified during the agent upgrade process. The file generated by the upgrade process is saved to the `/config` directory, to maintain your custom values, if any. You can then merge the two files with your customized values and save the resulting file in the `TWA_home/TWS` folder.

A template file containing default settings is located in *TWA_home*/TWS/config/localopts.

Note: All of the SSL settings in the localopts file relate to the network communications and do not relate to the Dynamic Workload Console.

The options in the `localopts` file are described in the following sections:

- "Localopts summary"
- "Localopts details" on page 37
- "Local options file example" on page 52

Localopts summary

General attributes of the workstation:

thiscpu = *workstation*
merge stdlists = *yes*|*no*
stdlist width = *columns*
syslog local = *facility*
restricted stdlists = *yes*|*no*

The attributes of the workstation for the batchman process:

bm check file = *seconds*
bm check status = *seconds*
bm look = *seconds*
bm read = *seconds*
bm stats = *on*|*off*
bm verbose = *on*|*off*
bm check until = *seconds*
bm check deadline = *seconds*
bm late every = *minutes*

The attributes of the workstation for the jobman process:

jm interactive old = *yes*|*no*
jm job table size = *entries*
jm load user profile = *on*|*off*
jm look = *seconds*
jm nice = *value*
jm promoted nice = *UNIX and Linux critical job priority*
jm promoted priority = *Windows critical job priority*

jm no root = *yes* | *no*
jm file no root = *yes* | *no*
jm read = *seconds*

The attributes of the workstation for the mailman process:

mm planoffset = *HHMM*
mm response = *seconds*
mm retrylink = *seconds*
mm sound off = *yes* | *no*
mm unlink = *seconds*
mm cache mailbox = *yes* | *no*
mm cache size = *bytes*
mm resolve master = *yes* | *no*
autostart monman = *yes* | *no*
mm read = *minutes*

The attributes of the workstation for the netman process:

nm mortal = *yes* | *no*
nm port = *port number*
nm read = *seconds*
nm retry = *seconds*

The attributes of the workstation for the writer process:

wr read = *seconds*
wr unlink = *seconds*
wr enable compression = *yes* | *no*

Optional attributes of the workstation for remote database files

mozart directory = *mozart_share*
parameters directory = *parms_share*
unison network directory = *unison_share*

The attributes of the workstation for the custom formats

date format = *integer*
composer prompt = *key*
conman prompt = *key*
switch sym prompt = *key*

The attributes of the workstation for the customization of I/O on mailbox files

sync level = *low* | *medium* | *high*

The attributes of the workstation for networking

tcp timeout = *seconds*
tcp connect timeout = *seconds*

The attributes of the workstation for SSL - General

ssl auth mode = *caonly* | *string* | *cpu*
ssl auth string = *string*
ssl fips enabled = *yes/no*
nm ssl full port = *value*
nm ssl port = *value*

OpenSSL attributes of the workstation - only used if *ssl fips enabled = "no"*

ssl key = **.pem*
ssl certificate = **.pem*
ssl key pwd = **.sth*
ssl ca certificate = **.crt*
ssl random seed = **.rnd*
ssl encryption cipher = *cipher*
cli ssl server auth = *yes | no*
cli ssl cipher = *string*
cli ssl server certificate = *file_name*
cli ssl trusted dir = *directory_name*
cli ssl tls10 cipher = *HIGH | <cipher>*
cli ssl tls11 cipher = *HIGH | <cipher>*
cli ssl tls12 cipher = *HIGH | <cipher>*
ssl tls10 cipher = *HIGH | <cipher>*
ssl tls11 cipher = *HIGH | <cipher>*
ssl tls12 cipher = *HIGH | <cipher>*

GSKit attributes of the workstation - only used if *ssl fips enabled = "yes"*

ssl keystore file = **.kdb*
ssl certificate keystore label = *name*
ssl keystore pwd = **.sth*
cli ssl keystore file = **.kdb*
cli ssl certificate keystore label = *name*
cli ssl keystore pwd = **.sth*
cli gsk tls10 cipher = *DFLT | <cipher>*
cli gsk tls11 cipher = *DFLT | <cipher>*
cli gsk tls12 cipher = *DFLT | <cipher>*
gsk tls10 cipher = *DFLT | <cipher>*
gsk tls11 cipher = *DFLT | <cipher>*
gsk tls12 cipher = *DFLT | <cipher>*

The attributes of the workstation for the WebSphere Application Server

local was = *yes | no*

Application server check attributes on the workstation

appserver check interval = *minutes*
appserver auto restart = *on | off*
appserver min restart time = *minutes*
appserver max restarts = *number*
appserver count reset interval = *hours*
appserver service name = *name*

The IBM Workload Scheduler instance is a command line client

is remote cli = *yes | no*

Attributes for command line client connection (conman)

host = *host_name*
protocol = *protocol*
port = *port number*
proxy = *proxy server*
proxy port = *proxy server port number*

time out = *seconds*
default ws = *master_workstation*
useropts = *useropts_file*

Note: The SSL attributes for the command line client connection will depend on which SSL method is in use. They are included in the relevant section and all commence with "cli".

Event Management parameters

can be event processor = *yes|no*

er load = *yes|no*

Centralized Agent Update parameters

DownloadDir = *directory_name*

Note: The `localopts` file syntax is not case-sensitive, and the spaces between words in the option names are ignored. For example, you can validly write **is remote cli** as:

- is remote cli
- Is Remote CLI
- isremotecli
- ISREMOTECLI
- isRemoteCLI
- ...

Localopts details

 # comment
Treats everything from the indicated sign (#) to the end of the line as a comment.

appserver auto restart = yes|no
Requests the `appservman` process to automatically start WebSphere Application Server if it is found down. The default is `Yes`.

appserver check interval = *minutes*
Specifies the frequency in minutes that the `appservman` process is to check that WebSphere Application Server is still running. The default is 5 minutes.

appserver count reset interval = *hours*
Specifies the time interval in hours after which the restart count is reset from the last WebSphere Application Server start. The default is 24 hours.

appserver max restarts = *number*
Specifies the maximum number of restarting attempts the `appservman` process can make before giving up and exiting without restarting WebSphere Application Server. The counter is reset if WebSphere Application Server runs for longer than the `appserver count reset interval` value. The default is 5.

appserver min restart time = *minutes*
Specifies in minutes the minimum elapsed time the `appservman` process must wait between each attempt to restart the WebSphere Application Server if it is down. If this value is less than the `appserver check interval`, the WebSphere Application Server is restarted as soon as it is

found down. If it is found down before this time interval (min restart time) has elapsed, `appservman` exits without restarting it. The default is 10 minutes.

appserver service name = *name*
Only in Windows environments. Specifies the name of the WebSphere Application Server windows service if different from the standard name. This field is generally not used.

autostart monman = yes|no
Used in event rule management. Restarts the monitoring engine automatically when the next production plan is activated (on Windows also when IBM Workload Scheduler is restarted). The default is `Yes`.

bm check deadline = *seconds*
Specify the minimum number of seconds Batchman waits before checking if a job has missed its deadline. The check is performed on all jobs and job streams included in the Symphony file, regardless of the workstation where the jobs and job streams are defined. Jobs and job streams with expired deadlines are marked as late in the local Symphony file. To obtain up-to-date information about the whole environment, define this option on the master domain manager. Deadlines for critical jobs are evaluated automatically, independently of the **bm check deadline** option. To disable the option and not check deadlines, enter a value of zero, the default value.

bm check file = *seconds*
Specify the minimum number of seconds Batchman waits before checking for the existence of a file that is used as a dependency. The default is 120 seconds.

bm check status = *seconds*
Specify the number of seconds Batchman waits between checking the status of an internetwork dependency. The default is 300 seconds.

bm check until = *seconds*
Specify the maximum number of seconds Batchman waits before reporting the expiration of an Until time for job or Job Scheduler. Specifying a value below the default setting (300) might overload the system. If it is set below the value of Local Option **bm read**, the value of **bm read** is used in its place. The default is 300 seconds.

= **bm look =** *seconds*
= Specify the minimum number of seconds Batchman waits before scanning
= and updating its production control file. If you install the 9.4, FP1 version
= as a fresh installation, the default value is automatically set to 5 for
= improving product performance. The previous default value was 15
= seconds and is maintained if you perform a product upgrade.

= **bm read =** *seconds*
= Specify the maximum number of seconds Batchman waits for a message in
= the `intercom.msg` message file. If no messages are in queue, Batchman
= waits until the timeout expires or until a message is written to the file. If
= you install the 9.4, FP1 version as a fresh installation, the default value is
= automatically set to 3 for improving product performance. The previous
= default value was 10 seconds and is maintained if you perform a product
= upgrade.

bm stats = on|off
To have Batchman send its startup and shut down statistics to its standard

list file, specify **on**. To prevent Batchman statistics from being sent to its standard list file, specify **off**. The default is **off**.

bm verbose = on|off
To have Batchman send all job status messages to its standard list file, specify **on**. To prevent the extended set of job status messages from being sent to the standard list file, specify **off**. The default is **off**.

bm late every = minutes
When an **every** job does not start at its expected start time, **bm late every** specifies the maximum number of minutes that elapse before IBM Workload Scheduler skips the job. This option applies only to jobs defined with **every** option together with the **at** time dependency, it has no impact on jobs that have only the **every** option.

can be event processor = yes|no
Specify if this workstation can act as event processing server or not. It is set by default to **yes** for master domain managers and backup masters, otherwise it is set to **no**.

cli gsk tls10 cipher=DFLT|<cipher>
Only used if SSL is defined using GSKit (`ssl fips enabled="yes"`). Specify the cipher to be used with the TLS 1.0 protocol in association with GSKit when using the IBM Workload Scheduler command line. Restart the agent to make the changes effective. This keyword is optional and must be manually inserted in the localopts file. When specified, it overrides the default option. If you set more parameters with different versions of the same protocol, the protocol with the lowest version is used.

cli gsk tls11 cipher=DFLT|<cipher>
Only used if SSL is defined using GSKit (`ssl fips enabled="yes"`). Specify the cipher to be used with the TLS 1.1 protocol in association with GSKit when using the IBM Workload Scheduler command line. Restart the agent to make the changes effective. This keyword is optional and must be manually inserted in the localopts file. When specified, it overrides the default option. If you set more parameters with different versions of the same protocol, the protocol with the lowest version is used.

cli gsk tls12 cipher=DFLT|<cipher>
Only used if SSL is defined using GSKit (`ssl fips enabled="yes"`). Specify the cipher to be used with the TLS 1.2 protocol in association with GSKit when using the IBM Workload Scheduler command line. Restart the agent to make the changes effective. This keyword is optional and must be manually inserted in the localopts file. When specified, it overrides the default option. If you set more parameters with different versions of the same protocol, the protocol with the lowest version is used.

cli ssl tls10 cipher=HIGH|<cipher>
Only used if SSL is defined using OpenSSL (`ssl fips enabled="no"`). Specify the cipher to be used with the TLS 1.0 protocol in association with SSL when using the IBM Workload Scheduler command line. Restart the agent to make the changes effective. This keyword is optional and must be manually inserted in the localopts file. When specified, it overrides the default option. If you set more parameters with different versions of the same protocol, the protocol with the lowest version is used.

cli ssl tls11 cipher=HIGH|<cipher>
Only used if SSL is defined using OpenSSL (`ssl fips enabled="no"`). Specify the cipher to be used with the TLS 1.1 protocol in association with SSL when using the IBM Workload Scheduler command line. Restart the

agent to make the changes effective. This keyword is optional and must be manually inserted in the localopts file. When specified, it overrides the default option. If you set more parameters with different versions of the same protocol, the protocol with the lowest version is used.

cli ssl tls12 cipher=HIGH|<cipher>
Only used if SSL is defined using OpenSSL (`ssl fips enabled="no"`). Specify the cipher to be used with the TLS 1.2 protocol in association with SSL when using the IBM Workload Scheduler command line. Restart the agent to make the changes effective. This keyword is optional and must be manually inserted in the localopts file. When specified, it overrides the default option. If you set more parameters with different versions of the same protocol, the protocol with the lowest version is used.

gsk tls10 cipher=DFLT|<cipher>
Only used if SSL is defined using GSKit (`ssl fips enabled="yes"`). Specify the cipher to be used with the TLS 1.0 protocol in association with GSKit. Restart the agent to make the changes effective. This keyword is optional and must be manually inserted in the localopts file. When specified, it overrides the default option. If you set more parameters with different versions of the same protocol, the protocol with the lowest version is used.

gsk tls11 cipher=DFLT|<cipher>
Only used if SSL is defined using GSKit (`ssl fips enabled="yes"`). Specify the cipher to be used with the TLS 1.1 protocol in association with GSKit. Restart the agent to make the changes effective. This keyword is optional and must be manually inserted in the localopts file. When specified, it overrides the default option. f you set more parameters with different versions of the same protocol, the protocol with the lowest version is used.

gsk tls12 cipher=DFLT|<cipher>
Only used if SSL is defined using GSKit (`ssl fips enabled="yes"`). Specify the cipher to be used with the TLS 1.2 protocol in association with GSKit. Restart the agent to make the changes effective. This keyword is optional and must be manually inserted in the localopts file. When specified, it overrides the default option. If you set more parameters with different versions of the same protocol, the protocol with the lowest version is used.

ssl tls10 cipher=HIGH|<cipher>
Only used if SSL is defined using OpenSSL (`ssl fips enabled="no"`). Specify the cipher to be used with the TLS 1.0 protocol in association with SSL. Restart the agent to make the changes effective. This keyword is optional and must be manually inserted in the localopts file. If you set more parameters with different versions of the same protocol, the protocol with the lowest version is used.

ssl tls11 cipher=HIGH|<cipher>
Only used if SSL is defined using OpenSSL (`ssl fips enabled="no"`). Specify the cipher to be used with the TLS 1.1 protocol in association with SSL. Restart the agent to make the changes effective. This keyword is optional and must be manually inserted in the localopts file. If you set more parameters with different versions of the same protocol, the protocol with the lowest version is used.

ssl tls12 cipher=HIGH|<cipher>
Only used if SSL is defined using OpenSSL (`ssl fips enabled="no"`). Specify the cipher to be used with the TLS 1.2 protocol in association with SSL. Restart the agent to make the changes effective. This keyword is

optional and must be manually inserted in the localopts file. If you set more parameters with different versions of the same protocol, the protocol with the lowest version is used.

cli ssl certificate keystore label = *string*
Only used if SSL is defined using GSKit (`ssl fips enabled="yes"`) Supply the label which identifies the certificate in the keystore when the command-line client is using SSL authentication to communicate with the master domain manager. The default is `IBM TWS 8.6 workstation`, which is the value of the certificate distributed with the product to all customers. This certificate is thus not secure and should be replaced with your own secure certificate. See "Configuring the SSL connection protocol for the network" on page 332.

cli ssl keystore file = *file_name*
Only used if SSL is defined using GSKit (`ssl fips enabled="yes"`). Specify the name of the keystore file used for SSL authentication when the command-line client is using SSL authentication to communicate with the master domain manager. The default is *TWA_home*`/TWS/ssl/TWSPublicKeyFile.pem`. This file is part of the SSL configuration distributed with the product to all customers. It is thus not secure and should be replaced with your own secure SSL configuration. See "Configuring the SSL connection protocol for the network" on page 332.

cli ssl keystore pwd = *file_name*
Only used if SSL is defined using GSKit (`ssl fips enabled="yes"`). Specify the password file of the keystore used for SSL authentication when the command-line client is using SSL authentication to communicate with the master domain manager. This file is part of the SSL configuration distributed with the product to all customers. It is thus not secure and should be replaced with your own secure SSL configuration. See "Configuring the SSL connection protocol for the network" on page 332.

cli ssl cipher = *cipher_class*
Only used if SSL is defined using OpenSSL (`ssl fips enabled="no"`) Specify the cipher class to be used when the command-line client and the server are using SSL authentication. Use one of the common cipher classes listed in Table 18. The default is MD5.

If you want to use an OpenSSL cipher class not listed in the table, use the following command to determine if your required class is supported:

```
openssl ciphers <class_name>
```

where *class_name* is the name of the class you want to use. If the command returns a cipher string, the class can be used.

Table 18. Valid encryption cipher classes

Encryption cipher class	Description
SSLv3	SSL version 3.0
TLS	Only for IBM Workload Scheduler, version 9.3. users: before enabling SSL communication, manually modify this value to **TLSv1**, as described in "Setting local options" on page 34. For users with V9.3 Fix Pack 1 or later, no manual intervention is required. The default value is TLSv1.
TLSv1	TLS version 1.0

Table 18. Valid encryption cipher classes (continued)

Encryption cipher class	Description
EXP	Export
EXPORT40	40-bit export
MD5	Ciphers using the MD5 digest, digital signature, one-way encryption, hash or checksum algorithm.
LOW	Low strength (no export, single DES)
MEDIUM	Ciphers with 128 bit encryption
HIGH	Ciphers using Triple-DES
NULL	Ciphers using no encryption

cli ssl server auth = yes|no
Only used if SSL is defined using OpenSSL (`ssl fips enabled="no"`) Specify **yes** if server authentication is to be used in SSL communications with the command line client. The default is **no**.

cli ssl server certificate = *file_name*
Only used if SSL is defined using OpenSSL (`ssl fips enabled="no"`) Specify the file, including its full directory path, that contains the SSL certificate when the command-line client and the server use SSL authentication in their communication. There is no default. See "Configuring the SSL connection protocol for the network" on page 332.

cli ssl trusted dir = *directory_name*
Only used if SSL is defined using OpenSSL (`ssl fips enabled="no"`) Specify the directory that contains an SSL trusted certificate contained in files with hash naming (#) when the command-line client and the server are using SSL authentication in their communication. When the directory path contains blanks, enclose it in double quotation marks ("). There is no default.

composer prompt = *prompt*
Specify the prompt for the composer command line. The prompt can be of up to 10 characters in length. The default is dash (**-**).

conman prompt = *prompt*
Specify the prompt for the conman command line. The prompt can be of up to 8 characters in length. The default is percent (**%**).

date format = 0|1|2|3
Specify the value that corresponds to the date format you require. The values can be:

- 0 corresponds to *yy/mm/dd*
- 1 corresponds to *mm/dd/yy*
- 2 corresponds to *dd/mm/yy*
- 3 indicates usage of Native Language Support variables

The default is **1**.

default ws = *manager_workstation*
The default workstation when you are accessing using a command line client. Specify the domain manager workstation.

DownloadDir = *directory_name*
Defines the name of the directory where the fix pack installation package

or upgrade eImage is downloaded during the centralized agent update process. If not specified, the following default directory is used:

On Windows operating systems:
<*TWA_home*>\TWS\stdlist\JM\download

On UNIX operating systems:
<*TWA_home*>/TWS/stdlist/JM/download

/ **er load = yes | no**
/ For UNIX and Linux operating systems only. If set to **yes**, specifies that the
/ IBM user profile should be loaded when running a GenericAction
/ EventRule. The default value is **no**.

host = *hostname_or_IP_address*
The name or IP address of the host when accessing using a command line client.

is remote cli = yes | no
Specify if this instance of IBM Workload Scheduler is installed as a command line client (yes).

jm interactive old = yes | no
Only for Windows operating systems starting from Vista and later versions. To comply with security restrictions introduced with the Vista version of Windows operating systems, only for fault-tolerant agents, IBM Workload Scheduler runs interactive jobs only if the `streamlogon` user has a valid, interactive session. Specify **yes** to allow **`jobman`** to start interactive jobs even if there are no active sessions for the `streamlogon` user. Specify **no** to allow **`jobman`** to start interactive jobs only if there are active sessions for the `streamlogon` user. The default is **no**.

jm job table size = *entries*
Specify the size, in number of entries, of the job table used by Jobman. The default is 1024 entries.

jm load user profile = on | off
Only on Windows operating systems. Specify if the **`jobman`** process loads the user profile and its environment variables for the user specified in the logon field of each job, before starting the job on the workstation. Specify **on** to load the user profile on the workstation before running jobs for the logon user; otherwise specify **off**. Roaming profiles are not supported. The default is **on**.

jm look = *seconds*
Specify the minimum number of seconds Jobman waits before looking for completed jobs and performing general job management tasks. The default is 300 seconds.

jm nice = *nice_value*
For UNIX and Linux operating systems only, specify the **nice** value to be applied to jobs launched by Jobman to change their priority in the kernel's scheduler. The default is zero.

The **nice** boundary values vary depending upon each specific platform, but generally lower values correspond to higher priority levels and vice versa. The default depends upon the operating system.

Applies to jobs scheduled by the root user only. Jobs submitted by any other user inherit the same **nice** value of the Jobman process.

See also jm promoted nice.

jm file no root = yes|no
For UNIX and Linux operating systems only, specify **yes** to prevent Jobman from executing commands in file dependencies as **root**. Specify **no** to allow Jobman to execute commands in file dependencies as **root**. The default is **no**.

jm no root = yes|no
For UNIX and Linux operating systems only, specify **yes** to prevent Jobman from launching **root** jobs. Specify **no** to allow Jobman to launch **root** jobs. The default is **yes**.

jm promoted nice = *nice_value*
Used in workload service assurance. For UNIX and Linux operating systems only, assigns the priority value to a critical job that needs to be promoted so that the operating system processes it before others. Applies to critical jobs or predecessors that need to be promoted so that they can start at their critical start time.

Boundary values vary depending upon each specific platform, but generally lower values correspond to higher priority levels and vice versa. The default is -1.

Be aware that:

- The promotion process is effective with negative values only. If you set a positive value, the system runs it with the -1 default value and logs a warning message every time Jobman starts.
- An out of range value (for example -200), prompts the operating system to automatically promote the jobs with the lowest allowed **nice** value. Note that in this case no warning is logged.
- Overusing the promotion mechanism (that is, defining an exceedingly high number of jobs as mission critical and setting the highest priority value here) might overload the operating system, negatively impacting the overall performance of the workstation.

You can use this and the jm nice options together. If you do, remember that, while **jm nice** applies only to jobs submitted by the root user, **jm promoted nice** applies only to jobs that have a critical start time. When a job matches both conditions, the values set for the two options add up. For example, if on a particular agent the local options file has:

```
jm nice= -2
jm promoted nice= -4
```

when a critical job submitted by the root user needs to be promoted, it is assigned a cumulative priority value of -6.

jm promoted priority = *value*
Used in workload service assurance. For Windows operating systems only, sets to this value the priority by which the operating system processes a critical job when it is promoted.

Applies to critical jobs or predecessors that need to be promoted so that they can start at their critical start time.

The possible values are:

- `High`
- `AboveNormal` (the default)
- `Normal`
- `BelowNormal`

- `Low` or `Idle`

Note that if you a set a lower priority value than the one non-critical jobs might be assigned, no warning is given and no mechanism like the one available for **jm promoted nice** sets it back to the default.

jm read = *seconds*
Specify the maximum number of seconds Jobman waits for a message in the `courier.msg` message file. The default is 10 seconds.

local was = yes|no
For master domain managers and backup masters connected to the IBM Workload Scheduler database. If set to **yes**, it improves the performance of Job Scheduler and job submission from the database The default is **no**.

merge stdlists = yes|no
Specify **yes** to have all of the IBM Workload Scheduler control processes, except Netman, send their console messages to a single standard list file. The file is given the name **TWSmerge**. Specify **no** to have the processes send messages to separate standard list files. The default is **yes**.

mm cache mailbox = yes|no
Use this option to enable Mailman to use a reading cache for incoming messages. In this case, only messages considered essential for network consistency are cached. The default is **yes**.

mm cache size = *messages*
Specify this option if you also use **mm cache mailbox**. The maximum value (default) is **512**.

mm planoffset = *HHMM*
HHMM is an amount of time in the format hours and minutes. When IBM Workload Scheduler starts, this amount of time is used as an offset to check the Symphony plan validity according to this formula:

```
current_timestamp < (Symphony_end_timestamp - HHMM)
```

If the result is true, that is, the current time is earlier than the Symphony planned end time minus the offset, the Symphony plan is considered valid and IBM Workload Scheduler starts. If the result is false, IBM Workload Scheduler does not start and an error is logged. The default for this optional attribute is an empty value; in this case, no check is performed by IBM Workload Scheduler on the validity of the plan. This check might be necessary when a domain manager stops because of an unplanned outage and restarts later, when a new domain manager has been started in the meanwhile, because not all the correct recovery procedures were run to exclude it from the IBM Workload Scheduler network. As a consequence, there are two domain managers running at the same time on the same fault-tolerant agent creating scheduling issues on all the fault-tolerant agents.

mm read = *seconds*
Specify the maximum number of seconds Mailman waits for a connection with a remote workstation. The default is 15 seconds.

mm resolve master = yes|no
When set to **yes** the $MASTER variable is resolved at the beginning of the production day. The host of any extended agent is switched after the next **`JnextPlan`** (long term switch). When it is set to **no**, the $MASTER variable is not resolved at **`JnextPlan`** and the host of any extended agent can be switched after a conman **switchmgr** command (short- and long-term

switch). The default is **yes**. When you switch a master domain manager and the original has **mm resolve master** set to no and the backup has **mm resolve master** set to yes, after the switch any extended agent that is hosted by $MASTER switches to the backup domain manager. When the backup domain manager restarts, the keyword $MASTER is locally expanded by Mailman. You should keep the **mm resolve master** value the same for master domain managers and backup domain managers.

mm response = *seconds*
Specify the maximum number of seconds Mailman waits for a response before reporting that a workstation is not responding. The minimum wait time for a response is **90** seconds. The default is 600 seconds.

mm retrylink = *seconds*
Specify the maximum number of seconds Mailman waits after unlinking from a non-responding workstation before it attempts to link to the workstation again. The default is 600 seconds. The **tomserver** optional mailman servers do not unlink non-responding agents. The link is repetitively checked every 60 seconds, which is the default **retrylink** for these servers.

mm sound off = yes|no
Specify how Mailman responds to a conman **tellop ?** command. Specify **yes** to have Mailman display information about every task it is performing. Specify **no** to have Mailman send only its own status. The default is **no**.

mm symphony download timeout = *seconds*
Specify the maximum number of minutes Mailman waits after attempting to initialize a workstation on a slow network. If the timeout expires without the workstation being initialized successfully, Mailman initializes the next workstation in the sequence. The default is no timeout (0).

mm unlink = *seconds*
Specify the maximum number of seconds Mailman waits before unlinking from a workstation that is not responding. The wait time should not be less than the response time specified for the Local Option **nm response**. The default is 960 seconds.

nm mortal = yes|no
Specify **yes** to have Netman quit when all of its child processes have stopped. Specify **no** to have Netman keep running even after its child processes have stopped. The default is **no**.

nm port = *port*
Specify the TCP port number that Netman responds to on the local computer. This must match the TCP/IP port in the computers workstation definition. It must be an unsigned 16-bit value in the range 1- 65535 (values between 0 and 1023 are reserved for services such as, FTP, TELNET, HTTP, and so on). The default is the value supplied during the product installation.

If you run event-driven workload automation and you have a security firewall, make sure this port is open for incoming and outgoing connections.

nm read = *seconds*
Specify the maximum number of seconds Netman waits for a connection request before checking its message queue for **stop** and **start** commands. The default is 10 seconds.

nm retry = *seconds*
Specify the maximum number of seconds Netman waits before retrying a connection that failed. The default is 800 seconds.

nm ssl full port = *port*
The port used to listen for incoming SSL connections when full SSL is configured by setting global option `enSSLFullConnection` to `yes` (see "Configuring full SSL security" on page 332 for more details). This value must match the one defined in the **secureaddr** attribute in the workstation definition in the database. It must be different from the **nm port** local option that defines the port used for normal communication.

Note:
1. If you install multiple instances of IBM Workload Scheduler on the same computer, set all SSL ports to different values.
2. If you plan not to use SSL, set the value to 0.

There is no default.

nm ssl port = *port*
The port used to listen for incoming SSL connections, when full SSL is not configured (see "Configuring full SSL security" on page 332 for more details). This value must match the one defined in the **secureaddr** attribute in the workstation definition in the database. It must be different from the **nm port** local option that defines the port used for normal communication.

Note:
1. If you install multiple instances of IBM Workload Scheduler on the same computer, set all SSL ports to different values.
2. If you plan not to use SSL, set the value to 0.

There is no default.

port = *port*
The TCP/IP port number of the protocol used when accessing using a command line client. The default is 31115.

protocol = http | https
The protocol used to connect to the host when accessing using a command line client.

proxy = *proxy_server_hostname_or_IP_address*
The name of the proxy server used when accessing using a command line client.

proxy port = *proxy_server_port*
The TCP/IP port number of the proxy server used when accessing using a command line client.

restricted stdlists = yes | no
Use this option to set a higher degree of security to the `stdlist` directory (and to its subdirectories) allowing only selected users to create, modify, or read files.

This option is available for UNIX workstations only. After you define it, make sure you erase your current `stdlist` directory (and subdirectories) and that you restart IBM Workload Scheduler. The default is `no`.

If the option is not present or if it is set to `no`, the newly created `stdlist` directory and its subdirectories are unaffected and their rights are as follows:

```
drwxrwxr-x  22 twsmdm staff          4096 Nov 09 12:12
drwxrwxr-x   2 twsmdm staff           256 Nov 09 11:40 2009.11.09
drwxrwxr-x   2 twsmdm staff          4096 Nov 09 11:40 logs
drwxr-xr-x   2 twsmdm staff          4096 Nov 09 11:40 traces
```

If the option is set to `yes`, these directories have the following rights:

```
drwxr-x--x  5 twsmdm staff            256 Nov 13 18:15
rwxr-x--x   2 twsmdm staff            256 Nov 13 18:15 2009.11.13
rwxr-x--x   2 twsmdm staff            256 Nov 13 18:15 logs
rwxr-x--x   2 twsmdm staff            256 Nov 13 18:15 traces
```

Do the following to define and activate this option:

1. Change the line `restricted stdlists = no` to `restricted stdlists = yes` in your local options file.
2. Stop IBM Workload Scheduler.
3. Stop WebSphere Application Server if present.
4. Remove the `stdlist` directory (or at least its files and subdirectories).
5. Start IBM Workload Scheduler.
6. Start WebSphere Application Server if present.

ssl auth mode = caonly | string | cpu

The behavior of IBM Workload Scheduler during an SSL handshake is based on the value of the SSL authentication mode option as follows:

caonly IBM Workload Scheduler checks the validity of the certificate and verifies that the peer certificate has been issued by a recognized CA. Information contained in the certificate is not examined. The default value.

string IBM Workload Scheduler checks the validity of the certificate and verifies that the peer certificate has been issued by a recognized CA. It also verifies that the Common Name (CN) of the Certificate Subject matches the string specified into the SSL auth string option. See “ssl auth string = string.”

cpu IBM Workload Scheduler checks the validity of the certificate and verifies that the peer certificate has been issued by a recognized CA. It also verifies that the Common Name (CN) of the Certificate Subject matches the name of the workstation that requested the service.

ssl auth string = *string*

Used in conjunction with the **SSL auth mode** option when the "string" value is specified. The **SSL auth string** (ranges from 1 — 64 characters) is used to verify the certificate validity. The default string is "tws".

ssl ca certificate = *file_name*

Only used if SSL is defined using OpenSSL (`ssl fips enabled="no"`) Specify the name of the file containing the trusted certification authority (CA) certificates required for SSL authentication. The CAs in this file are also used to build the list of acceptable client CAs passed to the client when the server side of the connection requests a client certificate. This file is the concatenation, in order of preference, of the various PEM-encoded CA certificate files.

The default is *TWA_home*`/TWS/ssl/TWSTrustedCA.crt`. This file is part of the SSL configuration distributed with the product to all customers. It is thus not secure and should be replaced with your own secure SSL configuration. See “Configuring the SSL connection protocol for the network” on page 332.

ssl certificate = *file_name*
Only used if SSL is defined using OpenSSL (`ssl fips enabled="no"`) Specify the name of the local certificate file used in SSL communication.

The default is *TWA_home*`/TWS/ssl/TWSPublicKeyFile.pem`. This file is part of the SSL configuration distributed with the product to all customers. It is thus not secure and should be replaced with your own secure SSL configuration. See "Configuring the SSL connection protocol for the network" on page 332.

ssl certificate keystore label = *string*
Only used if SSL is defined using GSKit (`ssl fips enabled="yes"`) Supply the label which identifies the certificate in the keystore when using SSL authentication.

The default is `IBM TWS 8.6 workstation`, which is the value of the certificate distributed with the product to all customers. This certificate is thus not secure and should be replaced with your own secure certificate. See "Configuring the SSL connection protocol for the network" on page 332.

ssl encryption cipher = *cipher_class*
Only used if SSL is defined using OpenSSL (`ssl fips enabled="no"`) Define the ciphers that the workstation supports during an SSL connection.

Use one of the common cipher classes listed in Table 18 on page 41. The default value is **TLSv1**. No manual intervention is required. If you want to use an OpenSSL cipher class not listed in the table, use the following command to determine if your required class is supported:

```
openssl ciphers <class_name>
```

where *class_name* is the name of the class you want to use. If the command returns a cipher string, the class can be used.

ssl fips enabled = yes|no
Determines whether your entire IBM Workload Scheduler network is enabled for FIPS (Federal Information Processing Standards) compliance. FIPS compliance requires the use of GSKit instead of the default OpenSSL for secure communications. If you enable FIPS (*ssl fips enabled="yes"*) you must set values for all the SSL attributes that apply to GSKit. If you do not enable FIPS (*ssl fips enabled="no"*), set the values for OpenSSL. The default is **no**. See "FIPS compliance" on page 339 for more details.

ssl key = *file_name*
Only used if SSL is defined using OpenSSL (`ssl fips enabled="no"`) The name of the private key file.

The default is *TWA_home*`/TWS/ssl/TWSPrivateKeyFile.pem`. This file is part of the SSL configuration distributed with the product to all customers. It is thus not secure and should be replaced with your own secure SSL configuration. See "Configuring the SSL connection protocol for the network" on page 332.

ssl key pwd = *file_name*
Only used if SSL is defined using OpenSSL (`ssl fips enabled="no"`) The name of the file containing the password for the stashed key.

The default is *TWA_home*`/TWS/ssl/TWSPrivateKeyFile.sth`. This file is part of the SSL configuration distributed with the product to all customers. It is

thus not secure and should be replaced with your own secure SSL configuration. See "Configuring the SSL connection protocol for the network" on page 332.

ssl keystore file = *file_name*
Only used if SSL is defined using GSKit (`ssl fips enabled="yes"`). Specify the name of the keystore file used for SSL authentication.

The default is *TWA_home*`/TWS/ssl/TWSKeyRing.kdb`. This file is part of the SSL configuration distributed with the product to all customers. It is thus not secure and should be replaced with your own secure SSL configuration. See "Configuring the SSL connection protocol for the network" on page 332.

ssl keystore pwd = *file_name*
Only used if SSL is defined using GSKit (`ssl fips enabled="yes"`). Specify the name of the keystore password file used for SSL authentication.

The default is *TWA_home*`/TWS/ssl/TWSKeyRing.sth`. This file is part of the SSL configuration distributed with the product to all customers. It is thus not secure and should be replaced with your own secure SSL configuration. See "Configuring the SSL connection protocol for the network" on page 332.

ssl random seed = *file_name*
Only used if SSL is defined using OpenSSL (`ssl fips enabled="no"`) Specify the pseudo random number file used by OpenSSL on some operating systems. Without this file, SSL authentication might not work correctly.

The default is *TWA_home*`/TWS/ssl/TWS.rnd`. This file is part of the SSL configuration distributed with the product to all customers. It is thus not secure and should be replaced with your own secure SSL configuration. See "Configuring the SSL connection protocol for the network" on page 332.

stdlist width = *columns*
Specify the maximum width of the IBM Workload Scheduler console messages. You can specify a column number in the range **1** to **255**. Lines are wrapped at or before the specified column, depending on the presence of imbedded carriage control characters. Specify a negative number or zero to ignore line width. On UNIX and Linux operating systems, you should ignore line width if you enable system logging with the **syslog local** option. The default is 0 columns.

switch sym prompt = *prompt*
Specify a prompt for the conman command line after you have selected a different Symphony file with the **setsym** command. The maximum length is 8 characters. The default is **n%**.

sync level = low | medium | high
Specify the rate at which IBM Workload Scheduler synchronizes information written to disk. This option affects all mailbox agents and is applicable to UNIX and Linux operating systems only. Values can be:

low Allows the operating system to handle it.

medium
Flushes the updates to disk after a transaction has completed.

high Flushes the updates to disk every time data is entered.

The default is **low**.

syslog local = *value*
Enables or disables IBM Workload Scheduler system logging for UNIX and Linux operating systems only. Specify **-1** to turn off system logging for IBM Workload Scheduler. Specify a number from **0** to **7** to turn on system logging and have IBM Workload Scheduler use the corresponding local facility (LOCAL0 through LOCAL7) for its messages. Specify any other number to turn on system logging and have IBM Workload Scheduler use the USER facility for its messages. The default is -1. See "IBM Workload Scheduler console messages and prompts" on page 97.

tcp connect timeout = *seconds*
Specify the maximum number of seconds that can be waited to establish a connection through non-blocking socket. The default is 15 seconds.

tcp timeout = *seconds*
Specify the maximum number of seconds that can be waited for the completion of a request on a connected workstation that is not responding. The default is 300 seconds.

this cpu = *workstation_name*
Specify the IBM Workload Scheduler name of this workstation. The name can be a maximum of 16 alphanumeric characters in length and must start with a letter. When a switch is made between the master domain manager and a backup domain manager, using the **switchmgr** command, the Symphony header value for **this cpu** is overwritten by the **this cpu** value in the `localopts` file. The default is the host name of the computer.

timeout = *seconds*
The timeout in seconds when accessing using a command line client. The default is 3600 seconds.

unison network directory = *directory_name*
This parameter applies only to versions of IBM Workload Scheduler prior to version 8.3. Defines the name of the Unison network directory. The default is `<TWA_home>/../unison/network`.

useropts = *file_name*
If you have multiple instances of IBM Workload Scheduler on a system, use this to identify the *useropts* file that is to be used to store the connection parameters for the instance in which this *localops* file is found. See "Multiple product instances" on page 56 for more details.

wr enable compression = yes | no
Use this option on fault-tolerant agents. Specify if the fault-tolerant agent can receive the Symphony file in compressed form from the master domain manager. The default is **no**.

wr read = *seconds*
Specify the number of seconds the Writer process waits for an incoming message before checking for a termination request from Netman. The default is 600 seconds.

wr unlink = *seconds*
Specify the number of seconds the Writer process waits before exiting if no incoming messages are received. The minimum is 120 seconds. The default is 180 seconds.

Local options file example

The following is an example of a default `localopts` file:

Note: Some parameters might not be present depending upon your version and configuration.

```
#################################################################
# Licensed Materials - Property of IBM* and HCL**
# 5698-WSH
# (C) Copyright IBM Corp. 1998, 2016 All rights reserved.
# (C) Copyright HCL Technologies Ltd. 2016 All rights reserved
# * Trademark of International Business Machines
# ** Trademark of HCL Technologies Limited
#################################################################
#
# The IBM Workload Scheduler localopts file defines the attributes of this
# workstation, for various processes.
#
#--------------------------------------------------------------------------
# General attributes of this workstation:
#
thiscpu=FTA_nc004163
merge stdlists      =yes
stdlist width       =0
syslog local        =-1
restricted stdlists =no
#
#--------------------------------------------------------------------------
# The attributes of this workstation for the batchman process:
#
bm check file      =120
bm check status    =300
bm look            =15
bm read            =10
bm stats           =off
bm verbose         =off
bm check until     =300
bm check deadline  =30
bm late every      =0
#
#--------------------------------------------------------------------------
# The attributes of this workstation for the jobman process:
#
jm job table size  =1024
jm look            =300
jm nice            =0
jm promoted nice   =-1    #UNIX
jm promoted priority =AboveNormal #WINDOWS
jm no root         =yes
jm file no root    =no
jm read            =10
#
#--------------------------------------------------------------------------
# The attributes of this workstation for the TWS mailman process:
#
mm response        =600
mm retrylink       =600
mm sound off       =no
mm unlink          =960
mm cache mailbox   =yes
mm cache size      =512
mm resolve master  =yes
autostart monman   =yes
mm symphony download timeout =0
#
#--------------------------------------------------------------------------
```

```
# The attributes of this workstation for the netman process:
#
nm mortal          =no
nm port            =35111
nm read            =10
nm retry           =800
#
#----------------------------------------------------------------------------
# The attributes of this workstation for the writer process:
#
wr read                =600
wr unlink              =180
wr enable compression  =no
#
#----------------------------------------------------------------------------
# Optional attributes of this Workstation for remote database files
#
# mozart directory =         /home/ITAuser/TWA/TWS/mozart
# parameters directory =     /home/ITAuser/TWA
# unison network directory = /home/ITAuser/TWA/TWS/../unison/network
#
#----------------------------------------------------------------------------
# The attributes of this workstation for custom formats
#
date format             =1 # The possible values are 0-yyyy/mm/dd, 1-mm/dd/yyyy,
2-dd/mm/yyyy, 3-NLS.
composer prompt         =-
conman prompt           =%
switch sym prompt       =<n>%
#
#----------------------------------------------------------------------------
# The attributes of this workstation for the customization of I/O on mailbox files
# sync level              =low
#
#----------------------------------------------------------------------------
# The attributes of this workstation for networking
# tcp timeout             =300 tcp connect timeout=5
#
#----------------------------------------------------------------------------
# General Secure options
#
SSL auth mode           =caonly
#
# Use "SSL auth string" only if "SSL auth mode" is set to "string"
#
SSL auth string     =tws
#
# The value "yes" for "SSL Fips enabled" forces TWS to use GSKIT,
else it uses OpenSSL
# This flag set to "yes" enables the FIPS 140-2 policies. The default value is "no".
#
SSL Fips enabled=yes
#
# Netman full SSL port, use "nm SSL full port"it on if "enSSLFullConnection"
is set to "yes"
# the value "0" means port close
#
nm SSL full port=0
#
# Netman SSL port
# the value "0" means port close
#
nm SSL port=33113
#
# End General Secure options
#----------------------------------------------------------------------------
```

```
#--------------------------------------------------------------------------
# OpenSSL option, TWS uses them if "SSL Fips enabled" is "no" ( the default )
#
SSL key="/home/ITAuser/TWA/TWS/ssl/OpenSSL/TWSClient.key"
SSL certificate="/home/ITAuser/TWA/TWS/ssl/OpenSSL/TWSClient.cer"
SSL key pwd="/home/ITAuser/TWA/TWS/ssl/OpenSSL/password.sth"
SSL CA certificate="/home/ITAuser/TWA/TWS/ssl/OpenSSL/
TWSTrustCertificates.cer"
SSL random seed="/home/ITAuser/TWA/TWS/ssl/OpenSSL/TWS.rnd"
SSL Encryption Cipher=TLSv1
CLI SSL cipher=HIGH
#
#CLI SSL server auth =
#CLI SSL server certificate =
#CLI SSL trusted dir =
# End OpenSSL options
#--------------------------------------------------------------------------

#--------------------------------------------------------------------------
# GSKIT options, TWS uses them if "SSL Fips enabled" is "yes"
#
SSL keystore file="/home/ITAuser/TWA/TWS/ssl/GSKit/TWSClientKeyStore.kdb"
SSL certificate keystore label="client"
SSL keystore pwd="/home/ITAuser/TWA/TWS/ssl/GSKit/TWSClientKeyStore.sth"
#
#
CLI SSL keystore file="/home/ITAuser/TWA/TWS/ssl/GSKit/
TWSClientKeyStore.kdb"
CLI SSL certificate keystore label="client"
CLI SSL keystore pwd="/home/ITAuser/TWA/TWS/ssl/GSKit/
TWSClientKeyStore.sth"
#----------------------- End GSKit options ---------------------------------

#--------------------------------------------------------------------------
# The TWS instance has been installed as REMOTE CLI
IS REMOTE CLI = no  # yes for a REMOTE CLI installation, no otherwise

#--------------------------------------------------------------------------
# Attributes for CLI connections
#
# General attributes for CLI connections
#
HOST=nc004113
PROTOCOL=https
PORT=35116
#PROXY            =
#PROXYPORT        =
#TIMEOUT          = 3600      # Timeout in seconds to wait a server response
#CLI SSL SERVER AUTH = yes

#DEFAULTWS        =
#USEROPTS          =

#--------------------------------------------------------------------------
# Event Management parameters
#
CAN BE EVENT PROCESSOR = no # yes for MDM and BKM, no otherwise

#--------------------------------------------------------------------------
# Centralized Agent Update
#
#DownloadDir =

SSL certificate chain      =/home/ITAuser/TWA/TWS/ssl/TWSCertificateChain.crt
merge logtrace      = yes
LOCAL WAS      = no
mm read      = 15
```

```
tcp connection timeout      = 15
#CLI SSL server auth      =

#CLI SSL server auth      =

#CLI SSL server auth      =
#CLI SSL server auth      =
#CLI SSL server auth      =



#CLI SSL server auth      =
#CLI SSL server auth      =
#CLI SSL server auth      =

#CLI SSL server auth      =
```

Note: The "REMOTE CLI" term indicates the command line client.

Setting user options

Set the user options you require for each user on a workstation who needs them in the `useropts` file. Changes do not take effect until IBM Workload Scheduler is stopped and restarted.

The concept of the `useropts` file is to contain values for `localopts` parameters that must be personalized for an individual user. The files must be located in the *user_home*`/.TWS` directory of the user. When IBM Workload Scheduler needs to access data from the `localopts` file, it looks first to see if the property it requires is stored only or also in the `useropts` file for the user, always preferring the `useropts` file version of the value of the key. If a property is not specified when invoking the command that requires it, or inside the `useropts` and `localopts` files, an error is displayed.

The main use of the `useropts` file is to store the user-specific connection parameters used to access the command line client (see "Configuring command-line client access authentication" on page 93). These are the following keys, which are not stored in the `localopts` file:

username
: User name used to access the master domain manager. The user must be defined in the security file on the master domain manager (see Chapter 4, "Configuring user authorization (Security file)," on page 183)

password
: Password used to access the master domain manager. The presence of the `ENCRYPT` label in the password field indicates that the specified setting has been encrypted; if this label is not present, you must exit and access the interface program again to allow the encryption of that field.

A `useropts` file is created for the <*TWS_user*> during the installation, but you must create a separate file for each user that needs to use user-specific parameters on a workstation.

Sample useropts file

This is the sample content of a `useropts` file:

```
#
# IBM Workload Scheduler useropts file defines attributes of this Workstation.
#
```

```
#------------------------------------------------------------------------
# Attributes for CLI connections
USERNAME    = MDMDBE4    # Username used in the connection
PASSWORD    = "ENCRYPT:YEE7cEZs+HE+mEHCsdNOfg==" # Password used in the connection
#HOST       =            # Master hostname used when attempting a connection.
PROTOCOL    = https      # Protocol used to establish a connection with the Master.
#PROTOCOL   = http       # Protocol used to establish a connection with the Master.
PORT        = 3111       # Protocol port
#PROXY      =
#PROXYPORT  =
TIMEOUT     = 120        # Timeout in seconds to wait a server response
#DEFAULTWS  =

CLI SSL keystore file                   = "$(install_dir)/ssl/MyTWSKeyRing.kdb"
CLI SSL certificate keystore label      = "client"
CLI SSL keystore pwd                    = "$(install_dir)/ssl/MyTWSKeyRing.sth"
```

The SSL configuration options for the command line client depend on the type of SSL implemented - here GSKit is assumed.

Note: The # symbol is used to comment a line.

Multiple product instances

Because IBM Workload Scheduler supports multiple product instances installed on the same computer, there can be more than one instance of the `useropts` file per user. This is achieved by giving the `useropts` files for a user different names for each instance.

In the `localopts` file of each instance the option named ***useropts*** identifies the file name of the `useropts` file that has to be accessed in the *user_home*`/.TWS` directory to connect to that installation instance.

This means that, for example, if two IBM Workload Scheduler instances are installed on a computer and the user `operator` is a user of both instances, you could define the `useropts` credentials as follows:

- In the `localopts` file of the *first* instance the local option *useropts* = `useropts1` identifies the *operator_home*`/.TWS/useropts1` file containing the connection parameters settings that user `operator` needs to use to connect to the *first* IBM Workload Scheduler instance.
- In the `localopts` file of the *second* IBM Workload Scheduler instance the local option *useropts* = `useropts2` identifies the *operator_home*`/.TWS/useropts2` file containing the connection parameters settings that user `operator` needs to use to connect to the *second* IBM Workload Scheduler instance.

Configuring the agent

The configuration settings of the agent are stored in the `JobManager.ini` file. To find out where this file is located, see "Where products and components are installed" on page 1.

In a distributed environment, if a gateway is configured to allow the master domain manager or dynamic domain manager to communicate with a dynamic agent located behind a network boundary, then the gateway configuration settings of the agent are contained in the `JobManagerGW.ini` file. This file is almost identical to the `JobManager.ini` file, however, only parameters in the [ITA], [Env], and [ResourceAdvisorAgent] sections require configuration. For these parameters, definitions are given for both the `JobManager.ini` and `JobManagerGW.ini` files.

These files are made up of many different sections. Each section name is enclosed between square brackets and each section includes a sequence of `variable = value` statements.

You can customize properties for the following:

- Event-driven workload automation properties
- Log properties
- Trace properties when the agent is stopped. You can also customize traces when the agent is running using the procedure described in "Configuring trace properties when the agent is running" on page 61.
- Native job executor
- Java™ job executor
- Resource advisor agent
- System scanner

The log messages are written in the following file:

On Windows operating systems:

<*TWA_home*>\TWS\stdlist\JM\JobManager_message.log

On UNIX and Linux operating systems:

<*TWA_home*>/TWS/stdlist/JM/JobManager_message.log

The trace messages are written in the following file:

On Windows operating systems:

- <*TWA_home*>\TWS\stdlist\JM\ITA_trace.log
- <*TWA_home*>\TWS\stdlist\JM\JobManager_trace.log
- <*TWA_home*>\TWS\JavaExt\logs\javaExecutor0.log

On UNIX and Linux operating systems:

- <*TWA_home*>/TWS/stdlist/JM/ITA_trace.log
- <*TWA_home*>/TWS/stdlist/JM/JobManager_trace.log
- <*TWA_home*>/TWS/JavaExt/logs/javaExecutor0.log

Logging information about job types with advanced options

You can use the `logging.properties` file to configure the logging process for job types with advanced options, with the exception of the Executable and Access Method job types.

The `logging.properties` file is located on the IBM Workload Scheduler for z/OS Agent, under *TWA_home*/TWS/JavaExt/cfg/logging.properties.

After installation, this file is as follows:

```
# Specify the handlers to create in the root logger
# (all loggers are children of the root logger)
# The following creates two handlers
handlers = java.util.logging.ConsoleHandler,
           java.util.logging.FileHandler

# Set the default logging level for the root logger
.level = INFO

# Set the default logging level for new ConsoleHandler instances
java.util.logging.ConsoleHandler.level = INFO

# Set the default logging level for new FileHandler instances
java.util.logging.FileHandler.level
    = ALL
```

```
java.util.logging.FileHandler.pattern
    = C:\TWA_home\TWS\JavaExt\logs\javaExecutor%g.log
java.util.logging.FileHandler.limit
    = 1000000
java.util.logging.FileHandler.count
    = 10

# Set the default formatter for new ConsoleHandler instances
java.util.logging.ConsoleHandler.formatter =
            java.util.logging.SimpleFormatter
java.util.logging.FileHandler.formatter =
            java.util.logging.SimpleFormatter

# Set the default logging level for the logger named com.mycompany
com.ibm.scheduling = INFO
```

You can customize:

- The logging level (from INFO to WARNING, ERROR, or ALL) in the following keywords:

  .level Defines the logging level for the internal logger.

  com.ibm.scheduling
  Defines the logging level for the job types with advanced options. To log information about job types with advanced options, set this keyword to ALL.

- The path where the logs are written, specified by the following keyword:
  `java.util.logging.FileHandler.pattern`

Not all the properties in the `JobManager.ini` and `JobManagerGW.ini` files can be customized. For a list of the configurable properties, see the following sections:

- "Configuring log message properties [JobManager.Logging.cclog]" on page 59.
- "Configuring trace properties when the agent is stopped [JobManager.Logging.cclog]" on page 60.
- "Configuring common launchers properties [Launchers]" on page 64.
- "Configuring properties of the native job launcher [NativeJobLauncher]" on page 65.
- "Configuring properties of the Java job launcher [JavaJobLauncher]" on page 68.
- "Configuring properties of the Resource advisor agent [ResourceAdvisorAgent]" on page 68.
- "Configuring properties of the System scanner [SystemScanner]" on page 70
- See the section about configuring properties of event-driven workload automation [EventDrivenWorkload] in *Scheduling End-to-end with z-centric Capabilities*.

Configuring general properties [ITA]

About this task

In the `JobManagerGW.ini` file, you can add some general properties to the following section:

`[ITA]`

You can add or modify the following properties:

ActionPollers
The number of the thread processes started on the gateway workstation to communicate with the broker server installed on the master domain

manager or dynamic domain manager. The default value is 1. Specify this value if you have more than 100 dynamic agents that communicate with the broker server installed on the master domain manager or dynamic domain manager by using the same gateway. Restart the agent after the property change.

http_proxy
The URL of the proxy configured in a distributed environment through which agents or gateways communicate to the broker server installed on the master domain manager or dynamic domain manager. The value is `https_proxy =http://<proxy_workstation>:<proxy_workstation_port>`, where:

- *<proxy_workstation>* is the fully qualified host name of the workstation where the proxy is configured.
- *<proxy_workstation_port>* is the port number of the workstation where the proxy is configured.

Restart the agent after the property change.

Configuring log message properties [JobManager.Logging.cclog]

About this task

To configure the logs, edit the [JobManager.Logging.cclog] section in the `JobManager.ini` file. This procedure requires that you stop and restart the IBM Workload Scheduler agent

The section containing the log properties is named:

```
[JobManager.Logging.cclog]
```

You can change the following properties:

JobManager.loggerhd.fileName
The name of the file where messages are to be logged.

JobManager.loggerhd.maxFileBytes
The maximum size that the log file can reach. The default is 1024000 bytes.

JobManager.loggerhd.maxFiles
The maximum number of log files that can be stored. The default is 3.

JobManager.loggerhd.fileEncoding
By default, log files for the agent are coded in UTF-8 format. If you want to produce the log in a different format, add this property and specify the required codepage.

JobManager.loggerfl.level
The amount of information to be provided in the logs. The value ranges from 3000 to 7000. Smaller numbers correspond to more detailed logs. The default is 3000.

JobManager.ffdc.maxDiskSpace
Exceeding this maximum disk space, log files collected by the first failure data capture mechanism are removed, beginning with the oldest files first.

JobManager.ffdc.baseDir
The directory to which log and trace files collected by the ffdc tool are copied. Default directory is *<TWA_home>*`\TWS\stdlist\JM\JOBMANAGER-FFDC`.

JobManager.ffdc.filesToCopy
Log and trace files (JobManager_message.log and JobManager_trace.log) collected by the ffdc tool located in <*TWA_home*>\TWS\stdlist\JM. For example, `JobManager.ffdc.filesToCopy = "/opt/IBM/TWA_<TWS_user>/TWS/stdlist/JM/JobManager_message.log" "/opt/IBM/TWA_<TWS_user>/TWS/stdlist/JM/JobManager_trace.log"`

When a message is logged (JobManager.ffdc.triggerFilter = JobManager.msgIdFilter) that has an ID that matches the pattern "AWSITA*E" (JobManager.msgIdFilter.msgIds = AWSITA*E), which corresponds to all error messages, then the log and trace files (JobManager.ffdc.filesToCopy = "/opt/IBM/TWA_<TWS_user>/TWS/stdlist/JM/JobManager_message.log" "/opt/IBM/TWA_<TWS_user>/TWS/stdlist/JM/JobManager_trace.log") are copied (JobManager.ffdc.className = ccg_ffdc_filecopy_handler) to the directory JOBMANAGER-FFDC (JobManager.ffdc.baseDir = /opt/IBM/TWA_<TWS_user>/TWS/stdlist/JM/JOBMANAGER-FFDC). If the files copied exceed 10 MB (JobManager.ffdc.maxDiskSpace = 10000000), then the oldest files are removed first (JobManager.ffdc.quotaPolicy = QUOTA_AUTODELETE).

After installing the z-centric agent or dynamic agent on Windows 2012, the `JobManager_message.log` might not be created. In this case, perform the following procedure:

1. Stop the agent.
2. Create a backup copy of `JobManager.ini`, and edit the original file by changing the row:

   ```
   JobManager.loggerhd.className = ccg_multiproc_filehandler
   ```

   to

   ```
   JobManager.loggerhd.className = ccg_filehandler
   ```
3. Restart the agent.

Configuring trace properties when the agent is stopped [JobManager.Logging.cclog]

How to configure the trace properties when the agent is stopped.

To configure the trace properties when the agent is stopped, edit the [JobManager.Logging] section in the `JobManager.ini` file and then restart the IBM Workload Scheduler agent.

The section containing the trace properties is named:

```
[JobManager.Logging.cclog]
```

You can change the following properties:

JobManager.trhd.fileName
The name of the trace file.

JobManager.trhd.maxFileBytes
The maximum size that the trace file can reach. The default is 1024000 bytes.

JobManager.trhd.maxFiles
The maximum number of trace files that can be stored. The default is 3.

JobManager.trfl.level
Determines the type of trace messages that are logged. Change this value to trace more or fewer events, as appropriate, or on request from IBM Software Support. Valid values are:

DEBUG_MAX
Maximum tracing. Every trace message in the code is written to the trace logs.

INFO All *informational*, *warning*, *error* and *critical* trace messages are written to the trace. The default value.

WARNING
All *warning*, *error* and *critical* trace messages are written to the trace.

ERROR
All *error* and *critical* trace messages are written to the trace.

CRITICAL
Only messages which cause the agent to stop are written to the trace.

The output trace (JobManager_trace.log) is provided in XML format.

After installing the z-centric agent or dynamic agent on Windows 2012, the `JobManager_trace.log` might not be created. In this case, perform the following procedure:

1. Stop the agent.
2. Create a backup copy of `JobManager.ini`, and edit the original file by changing the row:

   ```
   JobManager.trhd.className = ccg_multiproc_filehandler
   ```

   to

   ```
   JobManager.trhd.className = ccg_filehandler
   ```
3. Restart the agent.

Configuring trace properties when the agent is running

Use the **twstrace** command to set the trace on the agent when it is running.

Using the **twstrace** command, you can perform the following actions on the agent when it is running:

- "See command usage and verify version" on page 62.
- "Enable or disable trace" on page 62.
- Set the traces to a specific level, specify the number of trace files you want to create, and the maximum size of each trace file. See "Set trace information" on page 62.
- "Show trace information" on page 63.
- Collect trace files, message files, and configuration files in a compressed file using the command line. See "Collect trace information" on page 63.
- Collect trace files, message files, and configuration files in a compressed file using the Dynamic Workload Console. See the section about retrieving IBM Workload Scheduler agent traces from the Dynamic Workload Console in *Troubleshooting Guide*.

You can also configure the traces when the agent is not running by editing the [JobManager.Logging] section in the `JobManager.ini` file as described in Configuring the agent section. This procedure requires that you stop and restart the agent.

twstrace command

Use the **twstrace** command to configure traces, and collect logs, traces, and configuration files (ita.ini and jobManager.ini) for agents. You collect all the information in a compressed file when it is running without stopping and restarting it.

See command usage and verify version

To see the command usage and options, use the following syntax.

Syntax

twstrace -u | **-v**

Parameters

-u Shows the command usage.

-v Shows the command version.

Enable or disable trace

To set the trace to the maximum or minimum level, use the following syntax.

Syntax

twstrace -enable | **-disable**

Parameters

-enable
Sets the trace to the maximum level. The maximum level is **1000**.

-disable
Sets the trace to the minimum level. The minimum level is **3000**.

Set trace information

To set the trace to a specific level, specify the number of trace files you want to create, and the maximum size the trace files can reach, use the following syntax.

Syntax

twstrace [**-level** <*level_number*>] [**-maxFiles** <*files_number*>] [**-maxFileBytes** <*bytes_number*>]

Parameters

-level <*level_number*>
Sets the trace level. Specify a value in the range from 1000 to 3000, which is also the default value. Note that if you set this parameter to 3000, you have the lowest verbosity level and the fewest trace messages. To have a better trace level, with the most verbose trace messages and the maximum trace level, set it to **1000**.

-maxFiles *<files_number>*
Specify the number of trace files you want to create.

-maxFileBytes *<bytes_number>*
Set the maximum size in bytes that the trace files can reach. The default is **1024000** bytes.

Show trace information

To display the current trace level, the number of trace files, and the maximum size the trace files can reach, use the following syntax.

Syntax

twstrace -level | -maxFiles | -maxFileBytes

Parameters

-level
See the trace level you set.

-maxFiles
See the number of trace files you create.

-maxFileBytes
See the maximum size you set for each trace file

Sample

The example shows the information you receive when you run the following command:

```
twstrace -level -maxFiles -maxFileBytes
AWSITA176I The trace properties are: level="1000",
max files="3", file size="1024000".
```

Collect trace information

To collect the trace files, the message files, and the configuration files in a compressed file, use the following syntax.

Syntax

twstrace -getLogs [**-zipFile** *<compressed_file_name>*] [**-host** *<host_name>*] [**-protocol** {*http | https* } [**-port** *<port_number>*][**-iniFile** *<ini_file_name>*]

Parameters

-zipFile *<compressed_file_name>*
Specify the name of the compressed file that contains all the information, that is logs, traces, and configuration files (ita.ini and jobManager.ini) for the agent. The default is **logs.zip**.

-host *<host_name>*
Specify the host name or the IP address of the agent for which you want to collect the trace. The default is **localhost**.

-protocol *http|https*
Specify the protocol of the agent for which you are collecting the trace. The default is the protocol specified in the **.ini** file of the agent.

-port <*port_number*>
Specify the port of the agent. The default is the port number of the agent where you are running the command line.

-iniFile <*ini_file_name*>
Specify the name of the **.ini** file that contains the SSL configuration of the agent for which you want to collect the traces. If you are collecting the traces for a remote agent for which you customized the security certificates, you must import the certificate on the local agent and specify the name of the **.ini** file that contains this configuration. To do this, perform the following actions:

1. Extract the certificate from the keystore of the remote agent.
2. Import the certificate in a local agent keystore. You can create an ad hoc keystore whose name must be **TWSClientKeyStore.kdb**.
3. Create an **.ini** file in which you specify:
   - **0** in the **tcp_port** property as follows:
     `tcp_port=0`
   - The port of the remote agent in the **ssl_port** property as follows:
     `ssl_port=<ssl_port>`
   - The path to the keystore you created in Step 2 in the **key_repository_path** property as follows:
     `key_repository_path=<local_agent_keystore_path>`

Configuring common launchers properties [Launchers]

About this task

In the `JobManager.ini` file, the section containing the properties common to the different launchers (or executors) is named:

`[Launchers]`

You can change the following properties:

BaseDir
The installation path of the IBM Workload Scheduler agent.

CommandHandlerMinThreads
The default is **20**.

CommandHandlerMaxThreads
The default is **100**.

CpaHeartBeatTimeSeconds
The polling interval in seconds used to verify if the **agent** process is still up and running. If the agent process is inactive the product stops also the **JobManager** process. The default is **30**.

DirectoryPermissions
The access rights assigned to the agent for creating directories when running jobs. The default is **0755**. Supported values are UNIX-format entries in hexadecimal notation.

DownloadDir
The name of the directory where the fix pack installation package or upgrade eImage for fault-tolerant agents or dynamic agents is downloaded during the centralized agent update process. If not specified, the following default directory is used:

On Windows operating systems:
<*TWA_home*>\TWS\stdlist\JM\download

On UNIX operating systems:
<*TWA_home*>/TWS/stdlist/JM/download

The centralized agent update process doesn't apply to z-centric agents.

ExecutorsMaxThreads
The default is **400**.

ExecutorsMinThreads
The default is **38**.

FilePermissions
The access rights assigned to the agent for creating files when running jobs. The default is **0755**. Supported values are UNIX-format entries in hexadecimal notation.

MaxAge
The number of days that job logs are kept (in path *TWA_home*/TWS/stdlidst/JM) before being deleted. The default is **30**. Possible values range from a minimum of 1 day.

NotifierMaxThreads
The default is **5**.

NotifierMinThreads
The default is **3**.

SpoolDir
The path to the folder containing the jobstore and outputs. The default is:
value of BaseDir/stdlidst/JM

StackSizeBytes
The size of the operating system stack in bytes. The default is **DEFAULT**, meaning that the **agent** uses the default value for the operating system.

Configuring properties of the native job launcher [NativeJobLauncher]

About this task

In the JobManager.ini file, the section containing the properties of the native job launcher is named:
[NativeJobLauncher]

You can change the following properties:

AllowRoot
Applies to UNIX systems only. Specifies if the root user can run jobs on the agent. It can be true or false. The default is false. This property does not apply to IBM i.

CheckExec
If true, before launching the job, the agent checks both the availability and the execution rights of the binary file. The default is true.

DefaultWorkingDir
Specifies the working directory of native jobs. You can also specify the value for the working directory when creating or editing the job definition in the Workload Designer. When specified in the Workload Designer, this

value overrides the value specified for the **DefaultWorkingDir** property. If you do not specify any working directories, the *<TWS_home>*\bin directory is used.

JobUnspecifiedInteractive
Applies to Windows operating systems only. Specifies if native jobs are to be launched in interactive mode. It can be `true` or `false`. The default is `false`.

KeepCommandTraces
Set to `true` to store the traces of the method invocation for actions performed on a job definition, for example, when selecting from a picklist. These files are stored in the path `/opt/IBM/TWA_<TWS_user>/TWS/stdlist/JM/r3batch_cmd_exec`. The default setting is `false`.

KeepJobCommandTraces
Set to `true` to store the traces of the method invocation for actions performed on a job instance, for example, viewing a spool list. These files are stored in the .zip file of the job instance. The default setting is `true`.

LoadProfile
Applies to agents on Windows servers only. Specifies if the user profile is to be loaded. It can be `true` or `false`. The default is `true`.

MonitorQueueName
Specifies the name of the queue where the IBM i jobs are monitored. If you do not specify this property, the default queue (QBATCH) is used.

PortMax
The maximum range of the port numbers used by the task launcher to communicate with the Job Manager. The default is 0, meaning that the operating system assigns the port automatically.

PortMin
The minimum range of the port numbers used by the task launcher to communicate with the Job Manager. The default is 0, meaning that the operating system assigns the port automatically.

PostJobExecScriptPathName
The fully qualified path of the script file that you want to run when the job completes. By default, this property is not present in the `JobManager.ini` file. If you do not specify any file path or the script file doesn't exist, no action is taken.

This property applies to dynamic agent and z/OS agent. For details about running a script when a job completes, see *User's Guide and Reference*.

PromotedNice
Used in workload service assurance. This property is not supported on the Agent for z/OS.

For UNIX and Linux operating systems only, assigns the priority value to a critical job that needs to be promoted so that the operating system processes it before others. Applies to critical jobs or predecessors that need to be promoted so that they can start at their critical start time.

Boundary values vary depending upon each specific platform, but generally lower values correspond to higher priority levels and vice versa. The default is -1.

Be aware that:

- The promotion process is effective with negative values only. If you set a positive value, the system runs it with the -1 default value.
- An out of range value (for example -200), prompts the operating system to automatically promote the jobs with the lowest allowed nice value.
- Overusing the promotion mechanism (that is, defining an exceedingly high number of jobs as mission critical and setting the highest priority value here) might overload the operating system, negatively impacting the overall performance of the workstation.

PromotedPriority
Used in workload service assurance. This property is not supported on the Agent for z/OS.

For Windows operating systems only, sets to this value the priority by which the operating system processes a critical job when it is promoted. Applies to critical jobs or predecessors that need to be promoted so that they can start at their critical start time. Valid values are:

- `High`
- `AboveNormal` (the default)
- `Normal`
- `BelowNormal`
- `Low` or `Idle`

Note that if you a set a lower priority value than the one non-critical jobs might be assigned, no warning is given.

RequireUserName

When `true`, requires that you add the user name in the JSDL job definition.

When `false`, runs with the user name used by job manager, that is:

- *TWS_user* on UNIX and Linux systems
- The local system account on Windows systems

The default is `false`.

RunExecutablesAsIBMiJobs
If you set this property to `true`, you can define IBM i jobs as generic jobs without using the XML definition. Generic jobs are automatically converted to IBM i jobs. As a side effect, generic jobs cannot be run when this parameter is enabled (`RunExecutablesAsIBMiJobs=true`). There is no default value because this property is not listed in the `JobManager.ini` file after the agent installation.

If you set this property to `true`, ensure that the user you used to install the agent has been granted the `*ALLOBJ` special authority.

ScriptSuffix
The suffix to be used when creating the script files. It is:

`.cmd` For Windows

`.sh` For UNIX

VerboseTracing
Enables verbose tracing. It is set to `true` by default.

Configuring properties of the Java job launcher [JavaJobLauncher]

About this task

In the `JobManager.ini` file, the section containing the properties of the Java job launcher is named:

```
[JavaJobLauncher]
```

You can change the following properties:

JVMDir
The path to the virtual machine used to start job types with advanced options. You can change the path to another compatible Java virtual machine.

JVMOptions
The options to provide to the Java Virtual Machine used to start job types with advanced options. Supported keywords for establishing a secure connection are:

- htttps.proxyHost
- https.proxyPort

Supported keywords for establishing a non-secure connection are:

- Dhttp.proxyHost
- Dhttp.proxyPort

For example, to set job types with advanced options, based on the default JVM http protocol handler, to the unauthenticated proxy server called with name myproxyserver.mycompany.com, define the following option:

```
JVMOptions = -Dhttp.proxyHost=myproxyserver.mycompany.com
 -Dhttp.proxyPort=80
```

Configuring properties of the Resource advisor agent [ResourceAdvisorAgent]

About this task

In the `JobManager.ini` and `JobManagerGW.ini` files, the section containing the properties of the Resource advisor agent is named:

```
[ResourceAdvisorAgent]
```

You can change the following properties:

BackupResourceAdvisorUrls
The list of URLs returned by the IBM Workload Scheduler master in a distributed environment or by the dynamic domain manager either in a z/OS or in a distributed environment. The agent uses this list to connect to the master or dynamic domain manager.

CPUScannerPeriodSeconds
The time interval that the Resource advisor agent collects resource information about the local CPU. The default value is every 10 seconds.

FullyQualifiedHostname
The fully qualified host name of the agent. It is configured automatically at installation time and is used to connect with the master in a distributed

environment or with the dynamic domain manager in a z/OS or in a distributed environment. Edit only if the host name is changed after installation.

NotifyToResourceAdvisorPeriodSeconds
The time interval that the Resource advisor agent forwards the collected resource information to the Resource advisor. The default (and maximum value) is every 180 seconds.

ResourceAdvisorUrl

JobManager.ini
The URL of the master in a distributed environment, or of the dynamic domain manager in a z/OS or in a distributed environment, that is hosting the agent. This URL is used until the server replies with the list of its URLs. The value is `https://$(tdwb_server):$(tdwb_port)/JobManagerRESTWeb/JobScheduler/resource`, where:

$(*tdwb_server***)**
is the fully qualified host name of the master in a distributed environment or of the dynamic domain manager either in a z/OS or in a distributed environment.

$(*tdwb_port***)**
is the port number of the master in a distributed environment or of the dynamic domain manager either in a z/OS or in a distributed environment.

It is configured automatically at installation time. Edit only if the host name or the port number are changed after installation, or if you do not use secure connection (set to `http`). If you set the port number to zero, the resource advisor agent does not start. The port is set to zero if at installation time you specify that you will not be using the master in a distributed environment or the dynamic domain manager either in a z/OS or in a distributed environment.

In a distributed environment, if **-gateway** is set to either `local` or `remote`, then this is the URL of the dynamic agent workstation where the gateway resides and through which the dynamic agents communicate. The value is `https://$(tdwb_server):$(tdwb_port)/ita/JobManagerGW/JobManagerRESTWeb/JobScheduler/resource`, where:

$(*tdwb_server***)**
The fully qualified host name of the dynamic agent workstation where the gateway resides and through which the dynamic agent communicates with the dynamic workload broker.

$(*tdwb_port***)**
The port number of the dynamic agent workstation where the gateway resides and through which the dynamic agent communicates with the dynamic workload broker.

JobManagerGW.ini
In a distributed environment, if **-gateway** is set to `local`, then **ResourceAdvisorUrl** is the URL of the master or dynamic domain

manager. The value is `https://$(tdwb_server):$(tdwb_port)/JobManagerRESTWeb/JobScheduler/resource`, where:

$(*tdwb_server***)**
The fully qualified host name of the master or dynamic domain manager.

$(*tdwb_port***)**
The port number of the master or dynamic domain manager.

ScannerPeriodSeconds
The time interval that the Resource advisor agent collects information about all the resources in the local system other than CPU resources. The default value is every 120 seconds.

The resource advisor agent, intermittently scans the resources of the machine (computer system, operating system, file systems and networks) and periodically sends an update of their status to the master or dynamic domain manager either in a z/OS or in a distributed environment.

The CPU is scanned every `CPUScannerPeriodSeconds` seconds, while all the other resources are scanned every `ScannerPeriodSeconds` seconds. As soon as one of the scans shows a significant change in the status of a resource, the resources are synchronized with the master in a distributed environment or the dynamic domain manager either in a z/OS or in a distributed environment. The following is the policy followed by the agent to tell if a resource attribute has significantly changed:

- A resource is added or deleted
- A string attribute changes its value
- A CPU value changes by more than `DeltaForCPU`
- A file system value changes by more than `DeltaForDiskMB` megabytes
- A Memory value changes by more than `DeltaForMemoryMB` megabytes

If there are no significant changes, the resources are synchronized with the IBM Workload Scheduler master in a distributed environment or with thedynamic domain manager either in a z/OS or in a distributed environment every `NotifyToResourceAdvisorPeriodSeconds` seconds.

Configuring properties of the System scanner [SystemScanner]

About this task

In the `JobManager.ini` file, the section containing the properties of the System scanner is named:

`[SystemScanner]`

You can change the following properties:

CPUSamples
The number of samples used to calculate the average CPU usage. The default value is 3.

DeltaForCPU
The change in CPU usage considered to be significant when it becomes

higher than this percentage (for example, DeltaForCPU is 20 if the CPU usage changes from 10 percent to 30 percent). The default value is 20 percent.

DeltaForDiskMB
The change in use of all file system resources that is considered significant when it becomes higher than this value. The default value is 100 MB.

DeltaForMemoryMB
The change in use of all system memory that is considered significant when it becomes higher than this value. The default value is 100 MB.

Configuring environment variables [Env]

About this task

Add the section `[Env]` to the `JobManagerGW.ini` configuration file and insert the environment variables that you need in your dynamic scheduling environment.

Regular maintenance

Regular maintenance refers to the mechanisms that are used on your dynamic workstation agents to free up storage space and improve performance.

Unlike fault-tolerant agents where maintenance tasks must be performed manually using the **rmstdlist** utility command, you can have regular maintenance performed on your dynamic agent workstations to keep disk space under control by configuring the following parameters as appropriate.

Table 19. Agent configuration parameters. Configuration parameters for maintenance

File	Parameter	Description
JobManager.ini located in the path `TWA_home/TWS/ITA/cpa/config`	MaxAge	The number of days that job logs are kept (in path `TWA_home/TWS/stdlidst/JM`) before being deleted. The default is 2. Possible values range from a minimum of 1 day.
	JobManager.loggerhd.maxFileBytes	The maximum size that the log file can reach. The default is 1024000 bytes.
	JobManager.loggerhd.maxFiles	The maximum number of log files that can be stored in the `stdlist/JM` directory. The default is 3.
	JobManager.ffdc.maxDiskSpace	The maximum disk space reached, by the log files collected by the First Failure Data Capture tool, after which the oldest files are removed.
	JobManager.trhd.maxFileBytes	The maximum size that the log file can reach. The default is 1024000 bytes.
	JobManager.trhd.maxFiles	The maximum number of log files that can be stored. The default is 3.
logging.properties located in the path `TWA_home/TWS/JavaExt/cfg/` Logs related to jobs with advanced options.	java.util.logging.FileHandler.limit	The maximum amount to write log messages to a file. Default value is 1000000 (bytes)
	java.util.logging.FileHandler.count	The number of output files to cycle through. Default value is 10.

Configuring the dynamic workload broker server on the master domain manager and dynamic domain manager

About this task

You can perform these configuration tasks after completing the installation of your master domain manager, dynamic domain manager, and dynamic agents, and any time that you want to change or tune specific parameters in your environment.

The configuration parameters for the dynamic workload broker server are defined by default at installation time. You modify a subset of these parameters using the files that are created when you install dynamic workload broker. The following files are created in the path:

`TWA_home/TDWB/config`

ResourceAdvisorConfig.properties
Contains configuration information about the **Resource Advisor**. For more information, see "ResourceAdvisorConfig.properties file" on page 74.

JobDispatcherConfig.properties
Contains configuration information about the **Job Dispatcher**. For more information, see "JobDispatcherConfig.properties file" on page 76.

BrokerWorkstation.properties
Contains configuration information about the broker server. "BrokerWorkstation.properties file" on page 79

CLIConfig.properties
Contains configuration information for the dynamic workload broker command line. This file is described in *IBM Workload Scheduler: Scheduling Workload Dynamically*.

audit.properties
Contains options for configuring the auditing of events. This file is documented in the *IBM Workload Scheduler: Troubleshooting Guide*.

You can modify a subset of the parameters in these files to change the following settings:

- Heartbeat signal from the agents.
- Time interval for job allocation to resources
- Time interval for notifications on resources
- Polling time when checking the status of remote engine workstations
- Maximum number of results when allocating jobs to global resources
- Encryption of any passwords sent in the JSDL definitions
- Time interval for retrying the operation after a **Job Dispatcher** failure
- Time interval for retrying the operation after a client notification failure
- Archive settings for job data
- Job history settings
- Command line properties (see *IBM Workload Scheduler: Scheduling Workload Dynamically*)

The editable parameters are listed in the following sections. If you edit any parameters that are not listed, the product might not work. After modifying the files, you must stop and restart the IBM® WebSphere® server.

Maintaining the dynamic workload broker server on the master domain manager and dynamic domain manager

About this task

Because one dynamic workload broker server is installed with your master domain manager and dynamic domain manager, and one server with every backup manager, you have at least two servers present in your IBM Workload Scheduler network. The server running with the master domain manager is the only one active at any time. The servers installed in the backup managers are idle until you switch managers, and the server in the new manager becomes the active server (see "Starting, stopping, and displaying dynamic workload broker status" on page 445 for important information about this scenario). To have a smooth transition from one server to another, when you switch managers, it is important that you keep the same configuration settings in the `ResourceAdvisorConfig.properties` and `JobDispatcherConfig.properties` files in all your servers. When you make a change in any of these files of your running dynamic workload broker server, remember to apply the same change also in the dynamic workload broker server idling on your backup manager.

Some of the settings for the dynamic workload broker server are stored in the local **BrokerWorkstation.properties** file and also in the IBM Workload Scheduler database. When you switch to the backup master domain manager or dynamic domain manager, the dynamic workload broker server settings are automatically updated on the backup workstation. For more information about the **BrokerWorkstation.properties** file, see "BrokerWorkstation.properties file" on page 79.

Note: The database is automatically populated with the information from the active workstation, regardless of whether it is the manager or the backup workstation. For example, if you modify the dynamic workload broker server settings on the backup master domain manager or dynamic domain manager, this change is recorded in the database. When you switch back to the manager workstation, the change is applied to the master domain manager or dynamic domain manager and the related local settings are overwritten.

It is important that you also keep the data pertinent to every dynamic workload broker server up-to-date. If you change the host name or port number of any of your dynamic workload broker servers, use the `exportserverdata` and `importserverdata` commands from the dynamic workload broker command line to record these changes in the IBM Workload Scheduler database. For information about these commands, see *Scheduling Workload Dynamically*.

The database records for your workload broker workstations all have LOCALHOST as the host name of the workstation. Leave the record as-is. Do not replace LOCALHOST with the actual host name or IP address of the workstation. LOCALHOST is used intentionally to ensure that the jobs submitted from IBM Workload Scheduler are successfully sent to the new local dynamic workload broker when you switch the master domain manager or dynamic domain manager.

Enabling unsecure communication with the dynamic workload broker server

About this task

By default, the dynamic workload broker server uses secure communication. You might need to enable unsecure communication, even though this type of communication is not recommended.

To enable unsecure communication with the dynamic workload broker server, perform the following steps on the master domain manager:

1. Run the exportserverdata command located in *installation_directory*/TDWB/bin:
   `exportserverdata -dbUsr db_instance_admin -dbPwd db_instance_admin_pwd`
2. Open the resulting server.properties file in a flat-text editor.
3. Copy the following line:
   `https://hostname:port/JobManagerRESTWeb/JobScheduler`
4. Change the copied line by replacing **https** with **http**:
   `http://hostname:port/JobManagerRESTWeb/JobScheduler`

   The file now contains two lines specifying the connection mode, one line specifying the https mode and one line specifying the http mode.
5. Save the file.
6. Import the new data with the importserverdata command located in *installation_directory*/TDWB/bin:
   `importserverdata -dbUsr db_instance_admin -dbPwd db_instance_admin_pwd`

For more information about the exportserverdata and importserverdata commands, see *IBM Workload Scheduler: Scheduling Workload Dynamically*.

ResourceAdvisorConfig.properties file

The parameters in this file affect the following dynamic workload broker server settings:

- Heartbeat signal from the agents
- Time interval for job allocation to resources
- Time interval for notifications on resources
- Polling time when checking the status of remote engine workstations
- Maximum number of results when allocating jobs to global resources

You can modify the following parameters in the `ResourceAdvisorConfig.properties` file:

DatabaseCheckInterval
: Specifies the time interval within which the dynamic workload broker server checks the availability of the database. The default value is **180** seconds.

ResourceAdvisorURL
: Specifies the URL of the **Resource Advisor**.

RaaHeartBeatInterval
: Specifies the time interval within which the **Resource Advisor** expects a heartbeat signal from the dynamic agent. The default value is **200** seconds. After the maximum number of retries (specified in the **MissedHeartBeatCount** parameter) is exceeded, the **Resource Advisor**

reports the related computer as unavailable. In a slow network, you might want to set this parameter to a higher value. However, defining a higher value might delay the updates on the availability status of computer systems. If, instead, you decrease this value together with the value defined for the **NotifyToResourceAdvisorPeriodSeconds** parameter, this might generate network traffic and increase CPU usage when updating cached data. The value defined in this parameter must be consistent with the **NotifyToResourceAdvisorPeriodSeconds** parameter defined in the `JobManager.ini` file, which defines the time interval for each dynamic agent to send the heartbeat signal to the **Resource Advisor**.

MissedHeartBeatCount
Specifies the number of missed heartbeat signals after which the computer is listed as not available. The default value is 2. In a slow network, you might want to set this parameter to a higher value.

MaxWaitingTime
Specifies the maximum time interval that a job must wait for a resource to become available. If the interval expires before a resource becomes available, the job status changes to Resource Allocation Failure. The default value is 600 seconds. You can override this value for each specific job by using the **Maximum Resource Waiting Time** parameter defined in the Job Brokering Definition Console. For more information about the **Maximum Resource Waiting Time** parameter, see the Job Brokering Definition Console online help. If you set this parameter to -1, no waiting interval is applied for the jobs. If you set this parameter to 0, the **Resource Advisor** tries once to find the matching resources and, if it does not find any resource, the job changes to the ALLOCATION FAILED status. If you increase this value, all submitted jobs remain in WAITING status for a longer time and the **Resource Advisor** tries to find matching resources according to the value defined for the **CheckInterval** parameter.

CheckInterval
Specifies how long the **Resource Advisor** waits before retrying to find matching resources for a job that did not find any resource in the previous time slot. The default value is 60 seconds.

TimeSlotLength
Specifies the time slot interval during which the **Resource Advisor** allocates resources to each job. Jobs submitted after this interval has expired are considered in a new time slot. The default value is 15 seconds. The default value is adequate for most environments and should not be modified. Setting this parameter to a higher value, causes the **Resource Advisor** to assign resources to higher priority jobs rather than to lower priority jobs when all jobs are trying to obtain the same resource. It might also, however, cause the job resource matching processing to take longer and the resource state updates from agents to be slowed down. Setting this parameter to a lower value, causes the **Resource Advisor** to process the resource matching faster and, if you have a high number of agents with frequent updates, to update the resource repository immediately. If job requirements match many resources, lower values ensure a better load balancing. If most jobs use resource allocation, do not lower this value because the allocation evaluation requires many processing resources.

NotifyTimeInterval
Specifies the interval within which the **Resource Advisor** retries to send notifications on the job status to the **Job Dispatcher** after a notification

failed. The default value is 15 seconds. The default value is adequate for most environments and should not be modified.

MaxNotificationCount
Specifies the maximum number of attempts for the **Resource Advisor** to send notifications to the **Job Dispatcher**. The default value is 100. The default value is adequate for most environments and should not be modified.

ServersCacheRefreshInterval

Specifies with what frequency (in seconds) the Resource Advisor checks the list of active and backup dynamic workload broker servers for updates. This list is initially created when the master domain manager is installed, and after that it is updated every time a new backup master is installed and connected to the master domain manager database (the master domain manager and every backup master include also a dynamic workload broker server). When the Resource Advisor agents send their data about the resources discovered in each computer, they are able to automatically switch between the servers of this list, so that the dynamic workload broker server that is currently active can store this data in its Resource Repository. For this reason, the Resource Advisor agents must know at all times the list of all dynamic workload broker servers. The possible values range between 300 (5 minutes) and 43200 (12 hours). The default value is 600 seconds.

StatusCheckInterval
Specifies the time interval in seconds the Resource Advisor waits before polling for the status of a resource. For example this timeout applies when checking the status of a remote engine. The default value is 120 seconds.

After modifying the file, you must stop and restart the WebSphere Application Server.

JobDispatcherConfig.properties file

The parameters in this file affect the following settings for the dynamic workload broker server installed on a master domain manager or dynamic domain manager:

- Encryption of any passwords sent in the JSDL definitions
- Time interval for retrying the operation after a **Job Dispatcher** failure
- Time interval for retrying the operation after a client notification failure
- Archive settings for job data
- Job history settings
- Gateways and dynamic workload broker server connection settings.

After modifying the file, you must stop and restart the IBM WebSphere server.

During the upgrade from version 8.5.1 the values you set for the properties for version 8.5.1 are preserved. The default values for the properties for version 8.6 are different from those in version 8.5.1. If you want to use the version 8.6 defaults, change them manually.

In the `JobDispatcherConfig.properties` file, the following parameters are available:

DatabaseCheckInterval
Specifies the time interval within which the dynamic workload broker server checks the availability of the database. The default value is **180** seconds.

EnablePasswordEncryption
Specifies that any user passwords contained in the JSDL definitions are to be encrypted when the definitions are sent to the agents. The default is `true`. Setting this property to `false` forces the dynamic workload broker server to send the passwords in plain text. This applies to any password field.

RAEndpointAddress
Specifies the URL of the **Resource Advisor**.

JDURL
Specifies the URL of the **Job Dispatcher**.

FailQInterval
Specifies the numbers of seconds for retrying the operation after the following failures:

- Client notification.
- Allocation, Reallocate, Cancel Allocation requests to **Resource Advisor**.
- Any database operation failed for connectivity reasons.

The default value is 30 seconds. Increasing this value improves recovery speed after a failure but can use many system resources if the recovery operation is complex. For example, if the workload broker workstation is processing a new Symphony file, this operation might require some time, so you should set this parameter to a high value. If you are not using workload broker workstation, this parameter can be set to a lower value.

MaxCancelJobAttemptsCount
The maximum number of times the Job Dispatcher attempts to cancel a shadow job or a job running on a dynamic agent when a request to kill the job is made and the kill request cannot be immediately processed. The default is 1440 attempts. The Job Dispatcher attempts to cancel the job every 30 seconds for a maximum number of times specified by this parameter.

MaxNotificationCount
Specifies the maximum number of retries after a client notification failure. The default value is 1440. For example, if the workload broker workstation is processing a new Symphony file, this operation might require some time, so you should set this parameter to a high value. If you are not using the workload broker workstation, this parameter can be set to a lower value.

= **MoveHistoryDataFrequencyInMins**
= Specifies how often job data must be deleted. The unit of measurement is
= minutes. The default value is 60 minutes. Increasing this value causes the
= **Job Dispatcher** to check less frequently for jobs to be deleted. Therefore,
= the volume of jobs in the **Job Repository** might increase and all queries
= might take longer to complete. Dynamic workload broker servers with
= high job throughput might require lower values, while low job
= throughputs might require higher values.

SuccessfulJobsMaxAge
Specifies how long successfully completed or canceled jobs must be kept in

the **Job Repository** database before being archived. The unit of measurement is hours. The default value is 240 hours, that is ten days.

UnsuccessfulJobsMaxAge
Specifies how long unsuccessfully completed jobs or jobs in unknown status must be kept in the **Job Repository** database before being archived. The unit of measurement is hours. The default value is 720 hours, that is 30 days.

AgentConnectTimeout
Specifies the number of minutes that the dynamic workload broker server waits for a scheduling agent response after it first attempts to establish a connection with that agent. If the agent does not reply within this time, the server does not open the connection. Values range from 0 to 60 (use 0 to wait indefinitely). The default is 3.

AgentReadTimeout
Specifies the number of minutes that the dynamic workload broker server waits to receive data from established connections with a scheduling agent or a gateway. If no data arrives within the specified time, the server closes the connection with the agent. Values range from 0 to 60 (use 0 to wait indefinitely). The default is 3.

GatewayPollingTimeout
Add this parameter to specify the number of minutes that the gateway waits to receive data from established connections with a dynamic workload broker. If no data arrives within the specified time, the gateway closes the connection with the dynamic workload broker. Values range from 1 to 60. The default is 1 minute.

GatewayConnectionTimeout
Add this parameter to specify the number of seconds that the dynamic workload broker server waits for a gateway receiving data after the dynamic workload broker first attempts to send data to the gateway. If the gateway does not reply within this time, the dynamic workload broker does not open the connection. Values range from 1 to 60. The default is 10 seconds.

MaxNumberOfParallelGateways
Add this parameter to specify the number of gateways that dynamic workload broker server can manage without lack of performances. Values range from 3 to 100. The default is 3.

Note:

You can use this file to configure the product behavior when archiving job data. For more information about archive tables, see "Historical database tables created during installation" on page 80.

If an unexpected job workload peak occurs and a cleanup of the database is required earlier than the value you specified in the `MoveHistoryDataFrequencyInMins` parameter, you can use the **movehistorydata** command to perform a cleanup before the scheduled cleanup is performed.

BrokerWorkstation.properties file

If you need to make configuration changes to the broker server after the installation has completed, you can edit the `BrokerWorkstation.properties` file. The `BrokerWorkstation.properties` file contains the following configuration properties:

DomainManager.Workstation.Name
The name of the domain manager workstation.

DomainManager.Workstation.Port
The port of the domain manager workstation.

MasterDomainManager.Name
The name of the master domain manager workstation.

Broker.Workstation.Name
The name of the broker server in the IBM Workload Scheduler production plan. This name is first assigned at installation time.

MasterDomainManager.HostName
The host name of the master domain manager workstation.

MasterDomainManager.HttpsPort
The HTTPS port of the master domain manager workstation.

Broker.Workstation.Port
The port used by the broker server to listen to the incoming traffic (equivalent to the Netman port). It is first assigned at installation time. This port number must always be the same for all the broker servers that you define in your IBM Workload Scheduler network (one with the master domain manager and one with every backup master you install) to ensure consistency when you switch masters.

DomainManager.Workstation.Domain
The name of the domain where the broker server is registered.

Broker.AuthorizedCNs
The list of prefixes of common names authorized to communicate with the broker server. For more information about authorizing the connection to the dynamic domain manager, see "Customizing the SSL connection between a master domain manager and a dynamic domain manager or its backup by using your certificates" on page 323.

Broker.Workstation.Enable
A switch that enables or disables the broker server. The value can be `true` or `false`. The default value is `true`.

Set this value to `false` if you decide not to use a broker server. Not using the broker server means that you can submit jobs dynamically on the dynamic workload broker directly (using either the Dynamic Workload Console or the dynamic workload broker command line) without using the scheduling facilities of IBM Workload Scheduler.

Broker.Workstation.CpuType
The workstation type assigned to the broker server. Supported values are:

- master domain manager (master)
- backup master domain manager (fta)
- dynamic domain manager (fta, broker, agent)
- backup dynamic domain manager (fta, broker, agent)

Broker.Workstation.RetryLink
The number of seconds between consecutive attempts to link to the broker server. The default is 600.

Note that no SSL security is available for the connection between the master domain manager and the broker server. All the data between the two workstations is sent unencrypted. If this might cause a security risk in your environment, you can choose not to use the broker server functions, by setting `Broker.Workstation.Enable` to `false`.

Archiving job data

Job definitions created using the Job Brokering Definition Console or the Dynamic Workload Console are stored in the **Job Repository** database. The **Job Repository** database stores also the jobs created when the job definitions are submitted to the dynamic workload broker.

Job information is archived on a regular basis. By default, successful jobs are archived every 24 hours. Jobs in failed or unknown status are archived by default every 72 hours. Archived jobs are moved to historical tables in the **Job Repository**.

You can configure the time interval after which job data is archived using the following parameters:

- **MoveHistoryDataFrequencyInMins**
- **SuccessfulJobsMaxAge**
- **UnsuccessfulJobsMaxAge**
- **ArchivedJobsMaxAge**

These parameters are available in the JobDispatcherConfig.properties file, as described in "JobDispatcherConfig.properties file" on page 76. You can also use the **movehistorydata** command to perform a cleanup before the scheduled cleanup is performed.

Historical database tables created during installation

Database creation differs depending on the database vendor you are using. If you are using DB2®, two databases are created by default when the dynamic workload broker server is installed. If you are using Oracle, two schemas are created in the same database. The names for both the databases and the schemas are as follows:

IBMCDB (DB2)/ CDB (Oracle)
Contains Agent manager data. The name is fixed and cannot be changed.

TDWB
Contains dynamic workload broker data. You can change the name.

The following three historical tables are created during the installation process in the **TDWB** database. These tables are used to contain historical data about job instances.

JOA_JOB_ARCHIVES
Contains archived job instances. See Table 20 on page 81.

JRA_JOB_RESOURCE_ARCHIVES
Contains resource information related to a job. See Table 21 on page 81.

MEA_METRIC_ARCHIVES
Contains metrics collected for a job. See Table 22 on page 82.

To improve RDBMS performance, you can move data from the standard tables to historical tables on a regular basis. You can configure RDBMS maintenance using the JobDispatcherConfig.properties file. For more information, see "JobDispatcherConfig.properties file" on page 76. You can also use the **movehistorydata** command to move data to the historical tables and delete archived data.

Table 20. JOA_JOB_ARCHIVES database table

Column Name	DB2 Data Type	Oracle Data Type	Length	Nullable	Description
JOA_ID	CHAR () FOR BIT DATA	RAW	16	No	Contains the unique identifier of the job
JOA_START_TIME	TIMESTAMP	TIMESTAMP	26	Yes	The start time of the job, if started
JOA_END_TIME	TIMESTAMP	TIMESTAMP	26	Yes	The end time of the job, if ended
JOA_JSDL_INSTANCE	CLOB	CLOB		No	The JSDL (job definition), stored in binary format
JOA_SUBMIT_USERNAME	VARCHAR	VARCHAR2	120	No	The submitter
JOA_TIMEZONE	VARCHAR	VARCHAR2	40	Yes	Not used in this release
JOA_STATE	DECIMAL	NUMBER	2	No	The job state code
JOA_RETURN_CODE	DECIMAL	NUMBER	10	No	The job return code
JOA_SUBMIT_TIME	TIMESTAMP	TIMESTAMP	26	No	The submit time
JOA_NAME	VARCHAR	VARCHAR2	250	No	The job definition name
JOA_NAMESPACE	VARCHAR	VARCHAR2	250	Yes	The job definition namespace
JOA_ALIAS_NAME	VARCHAR	VARCHAR2	250	Yes	The job definition alias
JOA_SUBMITTER_TYPE	VARCHAR	VARCHAR2	80	Yes	The submitter type (for example, TDWB CLI, TDWB UI)
JOA_UPDATE_TIME	TIMESTAMP	TIMESTAMP	26	No	The last update timestamp of actual row

Table 21. JRA_JOB_RESOURCE_ARCHIVES database table

Column Name	DB2 Data Type	Oracle Data type	Length	Nullable	Description
JOA_ID	CHAR () FOR BIT DATA	RAW	16	No	Contains the unique identifier of the job
JRA_RESOURCE_NAME	VARCHAR	VARCHAR2	250	No	The resource internal name
JRA_RESOURCE_TYPE	VARCHAR	VARCHAR2	30	No	The resource type (for example, ComputerSystem, FileSystem, ...)
JRA_RESOURCE_GROUP	DECIMAL	NUMBER	5	No	The group code (grouping an allocation for actual job)
JRA_DISPLAY_NAME	VARCHAR	VARCHAR2	250	Yes	The displayed name
JRA_IS_TARGET	DECIMAL	NUMBER	1	No	A flag indicating the root resource (typically the ComputerSystem)

Table 22. MEA_METRIC_ARCHIVES database table

Column Name	DB2 Data Type	Oracle Data Type	Length	Nullable	Description
JOA_ID	CHAR () FOR BIT DATA	RAW	16	No	Contains the unique identifier of the job
MEA_NAME	VARCHAR	VARCHAR2	80	No	The metric name (for example, JOB_MEMORY_USAGE, JOB_CPU_USAGE, ...)
MEA_TYPE	CHAR	CHARACTER	10	No	The metric datatype (for example, DECIMAL, ...)
MEA_VALUE	VARCHAR	VARCHAR2	250	No	The metric value

Table 23 lists the status of jobs as stored in the historical tables in association with the job status available in the Dynamic Workload Console and in the command line, mapping them with the related options in the **movehistorydata** command.

Table 23. Job statuses in the historical tables

movehistorydata option	Job status in tables	Dynamic Workload Console status	Command line status
SuccessfulJobsMaxAge	43	Completed successfully	SUCCEEDED_EXECUTION
	44	Canceled	CANCELED
UnsuccessfulJobMaxAge	41	Resource allocation failed	RESOURCE_ ALLOCATION_FAILED
	42	Run failed	FAILED_EXECUTION
	45	Unknown	UNKNOWN
	46	Unable to start	NOT_EXECUTED

Configuring to schedule J2EE jobs

About this task

Using the dynamic workload broker component you can schedule J2EE jobs. To do this you must complete the following configuration tasks:

- Configure the J2EE executor on every agent on which you submit J2EE jobs.
- Configure the J2EE Job Executor Agent on an external WebSphere Application Server

Configuring the J2EE executor

About this task

To dynamically schedule J2EE jobs, you must configure the following property files on every agent on which you submit J2EE jobs:

- J2EEJobExecutorConfig.properties
- logging.properties
- soap.client.props

These files are configured with default values at installation time. The values that you can customize are indicated within the description of each file.

J2EEJobExecutorConfig.properties file: The path to this file is *TWA_home*/TWS/JavaExt/cfg/J2EEJobExecutorConfig.properties (*TWA_home*\TWS\JavaExt\cfg\J2EEJobExecutorConfig.properties) on the agent.

The keywords of this file are described in the following table:

Table 24. J2EEJobExecutorConfig.properties file keywords

Keyword	Specifies...	Default value	Must be customized
wasjaas.default	The path to the IBM WebSphere configuration file (wsjaas_client.conf) used to authenticate on the external WebSphere Application Server using JAAS security.	*TWA_home*/TWS/JavaExt/cfg/wsjaas_client.conf or *TWA_home*\TWS\JavaExt\cfg\wsjaas_client.conf	Optionally yes, if you move the file to the path you specify.
credentials.mycred	The credentials (ID and password) used to establish the SOAP connection to the external WebSphere Application Serverwhen using indirect scheduling (the password must be {xor} encrypted)	wasadmin,{xor}KD4sPjsyNjE\= (ID=wasadmin and password=wasadmin in {xor} encrypted format)	Yes, see "Running {xor} encryption on your password" on page 84 to learn how to encrypt your password.
connector.indirect	The name of the communication channel with WebSphere Application Server. Selecting an indirect invoker means that dynamic workload broker uses an existing WebSphere Application Server scheduling infrastructure that is already configured on a target external WebSphere Application Server. When creating the job definition, you can specify if you want to use a direct or indirect connector in the **J2EE Application** pane in the **Application** page in the Job Brokering Definition Console, or in the **invoker** element in the JSDL file. For more information about the Job Brokering Definition Console, see the online help.	A single line with the following values separated by commas: • indirect keyword • Name of the scheduler: sch/MyScheduler • soap keyword • Host name of the external WebSphere Application Server instance: washost.mydomain.com • SOAP port of the WebSphere Application Server instance: 8880 • Path to the soap.client.props file: *TWA_home*/TWS/JavaExt/cfg/soap.client.props • Credentials keyword: mycred	You must customize the following: • The scheduler name. Replace the sch/MyScheduler string with the JNDI name of the IBM WebSphere scheduler that you plan to use. • The host name of the external WebSphere Application Server instance. • The SOAP port of the external WebSphere Application Server instance.

Table 24. J2EEJobExecutorConfig.properties file keywords (continued)

Keyword	Specifies...	Default value	Must be customized

connector.direct	The name of the direct communication channel without using the WebSphere Application Server scheduler. Select a direct invoker to have dynamic workload brokerimmediately forward the job to the external WebSphere Application Server instance components (EJB or JMS). When creating the job definition, you can specify if you want to use a direct or indirect connector in the **J2EE Application** pane in the **Application** page in the Job Brokering Definition Console, or in the **invoker** element in the JSDL file. For more information about the Job Brokering Definition Console, see the online help.	A single line with the following values separated by commas:
		• `direct` keyword
		• The following string:
		`com.ibm.websphere.naming.WsnInitialContextFactory`
		• The following string:
		`corbaloc:iiop:washost.mydomain.com:2809`
			You must customize the following:
			• The host name of the external WebSphere Application Server instance: `washost.mydomain.com`
			• The RMI port of the external WebSphere Application Server instance: `2809`

trustStore.path	The path to the WebSphere Application Server trustStore file (this file must be copied to this local path from the WebSphere Application Server instance).	*TWA_home*/TWS/JavaExt/cfg/DummyClientTrustFile.jks	You can change the path (*TWA_home*/TWS/JavaExt/cfg), if you copy the trustStore path from the external WebSphere Application Server to this path.
trustStore.password	The password for the WebSphere Application Server trustStore file.	WebAs	Yes

Running {xor} encryption on your password:
About this task

To {xor} encrypt your password, use the `PropFilePasswordEncoder` command located in the *WAS_home*/bin directory of the external WebSphere Application Server.

Follow these steps:

1. Open a new text file and write the following line:
   string=your_password_in_plain_text
2. Save the file with a *file_name* of your choice.
3. Run `PropFilePasswordEncoder` as follows:
   `PropFilePasswordEncoder` *file_name string*

   where:

   file_name
   Is the name of the file with your password.

string Is the *string* you used in the text file. This can be any word that you choose, for example, `password`, `mypwd`, `joe`, and so on.

4. When the command completes, open the text file again. The content has changed to:

   string={xor}*your_encrypted_password*

5. Copy your encrypted password, inclusive of the {xor} characters, and paste it where required into your property files.

For example, if you want to encrypt your password `catamaran`. Proceed as follows:

1. Open a text file and write the following:

   ```
   mypasswd=catamaran
   ```

2. Save the file with name `encrfile.txt`.
3. Run:

   ```
   PropFilePasswordEncoder encrfile.txt mypasswd
   ```

4. Open `encrfile.txt`. You find:

   ```
   mypasswd={xor}PD4rPjI+LT4x
   ```

5. Copy {xor}PD4rPjI+LT4x and paste it where you need to.

The logging.properties file:
About this task

The path to this file is *TWA_home*/TWS/JavaExt/cfg/logging.properties (*TWA_home*\TWS\JavaExt\cfg\logging.properties) on the agent.

After installation, this file is as follows:

```
# Specify the handlers to create in the root logger
 # (all loggers are children of the root logger)
 # The following creates two handlers
 handlers = java.util.logging.ConsoleHandler, java.util.logging.FileHandler

 # Set the default logging level for the root logger
 .level = INFO

 # Set the default logging level for new ConsoleHandler instances
 java.util.logging.ConsoleHandler.level = INFO

 # Set the default logging level for new FileHandler instances
 java.util.logging.FileHandler.level = ALL
 java.util.logging.FileHandler.pattern =
  C:\TWA_home\TWS\JavaExt\logs\javaExecutor%g.log
 java.util.logging.FileHandler.limit = 1000000
 java.util.logging.FileHandler.count = 10

 # Set the default formatter for new ConsoleHandler instances
 java.util.logging.ConsoleHandler.formatter = java.util.logging.SimpleFormatter
 java.util.logging.FileHandler.formatter = java.util.logging.SimpleFormatter

 # Set the default logging level for the logger named com.mycompany
 com.ibm.scheduling = INFO
```

You can customize:

- The logging level (from INFO to WARNING, ERROR, or ALL) in the following keywords:

  .level Defines the logging level for the internal logger.

com.ibm.scheduling
Defines the logging level for the job types with advanced options. To log information about job types with advanced options, set this keyword to ALL.

- The path where the logs are written, specified by the following keyword:
  `java.util.logging.FileHandler.pattern`

The soap.client.props file:
About this task

The path to this file is *TWA_home*/TWS/JavaExt/cfg/soap.client.props (*TWA_home*\TWS\JavaExt\cfg\soap.client.props) on the agent.

After installation, this file is as follows:

```
#-----------------------------------------------------------------------------
 # SOAP Client Security Enablement
 #
 # - security enabled status  ( false[default], true  )
 #-----------------------------------------------------------------------------
 com.ibm.SOAP.securityEnabled=false

 com.ibm.SOAP.loginUserid=wasadmin
 com.ibm.SOAP.loginPassword={xor}KD4sPjsyNjE\=

 #-----------------------------------------------------------------------------
 # SOAP Login Prompt
 #
 # The auto prompting will happen only if all of the following are met:
 #
 # - Running from a SOAP client
 # - Server is reachable and server security is enabled
 # - Username and password are not provided either on command line or in this
 #   file
 # - com.ibm.SOAP.loginSource below is set to either "stdin" or "prompt"
 #
 #   stdin: prompt in command window
 #   prompt: GUI dialog box; falls back to stdin if GUI not allowed
 #
 #   (So to disable auto prompting, set loginSource to nothing)
 #-----------------------------------------------------------------------------
 com.ibm.SOAP.loginSource=prompt

 #-----------------------------------------------------------------------------
 # SOAP Request Timeout
 #
 # - timeout (specified in seconds [default 180], 0 implies no timeout)
 #
 #-----------------------------------------------------------------------------
 com.ibm.SOAP.requestTimeout=180

 #-----------------------------------------------------------------------------
 # SSL configuration alias referenced in ssl.client.props
 #-----------------------------------------------------------------------------
 com.ibm.ssl.alias=DefaultSSLSettings
```

If you want to enable SOAP client security, you must:

1. Change `com.ibm.SOAP.securityEnabled` to `true`
2. Customize:
   - `com.ibm.SOAP.loginUserid` with the true WebSphere Application Server administrator user ID.

- `com.ibm.SOAP.loginPassword` with the true WebSphere Application Server administrator password in {xor} encrypted format. See "Running {xor} encryption on your password" on page 84.

Configuring the J2EE Job Executor Agent

About this task

To set up the environment on the external WebSphere Application Server, Version 7.0 for the J2EE Job Executor Agent, do the following:

Create a Service Integration Bus

1. Open the WebSphere Administrative Console (for example, `http://localhost:9060/admin`, depending on the admin port you configured).
2. Expand **Service Integration** and select **Buses**. The Buses window is displayed.
3. Click **New** to display the Buses configuration window.
4. Type a name for the new bus, for example **MyBus** and click **Next** and then **Finish** to confirm.
5. Click the MyBus name and the MyBus properties are displayed.
6. Under Topology, click **Bus Members**. The Buses→MyBus→Bus members window is displayed.
7. Click **Add**, select the **Server** radio button, choose **<your_application_server_name>**, click **Next**, and then click **Finish**.
8. When the `Confirm the addition of a new bus member` panel is displayed, click **Finish**.
9. Select **Service Integration → Buses → MyBus → Destinations → New**.
10. Select **Queue** as the type and click **Next**
11. Type **BusQueue** as the identifier and assign the queue to a bus member. Click **Next**. In the confirmation panel click **Finish**.

Configure the Default Messaging Service

1. From the left panel of the WebSphere Administrative Console, expand **Resources→ JMS→ JMS Providers**, then click **Default messaging** at the server level as scope.
2. In the **Connection Factories** section, click **New**.
3. On the New JMS connection factory window, type in the following fields:

   Name MyCF

   JNDI name
   jms/MyCF

   Bus name
   MyBus

   Provider endpoints
   <hostname>:<Basic SIB port number>:BootstrapBasicMessaging;<hostname>:<Secure SIB port number>:BootstrapSecureMessaging,

   where, <Basic SIB port number> and <Secure SIB port number> can be found by expanding **Servers**, selecting **<your_application_server_name>**, and then selecting **Messaging engine inbound transports** under **Server Messaging**.

4. Select again **Resources -→ JMS-→ JMS Providers → Default Messaging** at the server level as scope, locate the section **Destinations**, and click **Queues**. Click **New** and type in the following fields as shown:
   Name=MyQueue
   JNDI name=jms/MyQueue
   Bus name=MyBus
   Queue name=BusQueue

   Click **Ok**.
5. Select again **Resources → JMS → JMS Providers → Default Messaging** at the server level as scope, and locate the section **Activation Specifications**.
6. Click **JMS activation specification**. Click **New** and type in the following fields as shown:
   Name=MyActSpec
   JNDI name=eis/MyActSpec
   Bus name=MyBus
   Destination type=Queue
   Destination JNDI name=jms/MyQueue

   Click **Ok**.

Configure the Java security

1. Select **Security → Secure Administration, applications and infrastructure**.
2. Locate the **Authentication** section, expand the **Java Authentication and Authorization Service**, and click **J2C authentication data**.
3. Click **New** and type in the following fields as shown:
   Alias=*usr*
   User ID=*usr*
   Password=*pwd*

   where *usr* is the user ID authenticated when using connector security and *pwd* is the related password.
4. Click **Ok**.

Create an XA DataSource

1. In the left pane, go to **Resources → JDBC..→ JDBCProviders**. In the resulting right pane, check that the scope is pointing to **<your_application_server_name>**.
2. Locate the **DERBY JDBC Provider (XA)** entry and click it.
3. Locate the **Additional Properties** section and click **Data Sources**.
4. Click **New** and type in the following fields as shown:
   Name = MyScheduler XA DataSource
   JNDI name = jdbc/SchedulerXADS
   Database name = ${USER_INSTALL_ROOT}/databases/Schedulers/${SERVER}/SchedulerDB;create=true
5. At the top of the page, click **Test connection button**.
6. Even if you get a negative result, modify the **Database name** field, deleting the part `;create=true`. Click **Ok**.

Create a WorkManager

1. In the left pane, go to **Resources → Asynchronous beans → Work managers** and click **New**.
2. type in the following fields as shown:
   Name=SchedulerWM

JNDI name=wm/SchedulerWM

3. Click **Ok**.

Create and configure a scheduler

1. In the left pane, go to **Resources → Schedulers** and click **New**.
2. type in the following fields as shown:
   Name=MyScheduler
   JNDI name=sch/MyScheduler
   Data source JNDI name=jdbc/SchedulerXADS
   Table prefix=MYSCHED
   Work managers JNDI name=wm/SchedulerWM
3. Click **Ok**.
4. Select **MyScheduler** and click **Create tables**.
5. Deploy the test application.

Security order of precedence used for running J2EE tasks

There are three ways of verifying that a task runs with the correct user credentials. Tasks run with specified security credentials using the following methods:

1. Java Authentication and Authorization Service (JAAS) security context on the thread when the task was created.
2. `setAuthenticationAlias` method on the `TaskInfo` object.
3. A specified security identity on a `BeanTaskInfo` task `TaskHandler` EJB method.

The authentication methods are performed in the order listed above, so that if an authentication method succeeds, the following checks are ignored. This means that the *usr* and *pwd* credentials defined in **Configure the Java security** take precedence over any credentials specified in the tasks themselves.

Configuring to schedule job types with advanced options

About this task

In addition to defining job types with advanced options using the Dynamic Workload Console or the **composer** command, you can use the related configuration files. The options you define in the configuration files apply to all job types with advanced options of the same type. You can override these options when defining the job using the Dynamic Workload Console or the **composer** command.

Configuration files are available on each dynamic agent in TWA_home/TWS/JavaExt/cfg for the following job types with advanced options:

Table 25. Configuration files for job types with advanced options

Job type	File name	Keyword
• Database job type • MSSQL Job	DatabaseJobExecutor.properties	Use the `jdbcDriversPath` keyword to specify the path to the JDBC drivers. Define the keyword so that it points to the JDBC jar files directory, for example: `jdbcDriversPath=c:\\mydir\\jars\\jdbc` The JDBC jar files must be located in the specified directory or its subdirectories. Ensure you have list permissions on the directory and its sub subdirectories. **Note:** For the MSSQL database, use version 4 of the JDBC drivers.
Java job type	JavaJobExecutor.properties	Use the `jarPath` keyword to specify the path to the directory where the jar files are stored. This includes all jar files stored in the specified directory and all sub directories.
J2EE job type	J2EEJobExecutorConfig.properties	For more information about the J2EE job type, see Configuring to schedule J2EE jobs.

Configuring security roles for users and groups

At Dynamic Workload Console installation time, new predefined roles and groups are created in Integrated Solutions Console. These roles determine which Dynamic Workload Console windows are available to a user, and therefore which activities the user is authorized to perform from the Dynamic Workload Console. If you do not assign a role to a Integrated Solutions Console user, that user does not see any entry for dynamic workload broker in the navigation tree. Access to the entries in the navigation tree does not mean that the user can access product functions. There is a second level of authorization, which is determined by a set of security roles created at master domain manager installation time in WebSphere Application Server. These roles define the levels of authorization needed to perform product functions regardless of the interface used.

You must therefore map the users and roles in WebSphere Application Server to match the users and roles defined in the Integrated Solutions Console so that communication between the Dynamic Workload Console and the dynamic workload broker instance is ensured. This procedure is described in "Mapping security roles to users and groups in WebSphere Application Server"

Mapping security roles to users and groups in WebSphere Application Server

About this task

When the dynamic workload broker instance is installed on your master domain manager, corresponding roles are set up in WebSphere Application Server. By default, these roles are not used. However, if you enable global security in your environment, the authorization required to perform any tasks is always validated by WebSphere Application Server. Users are required to provide credentials for accessing dynamic scheduling tasks. These credentials correspond to existing users defined in the domain user registry or the LDAP server.

To allow users and groups to access the dynamic workload broker functions when global security is enabled, they must be mapped to the security roles in WebSphere

Application Server. This mapping allows those users and groups to access applications defined by the role. At installation time, the following actor roles are created in the WebSphere Application Server:

Operator
Monitors and controls the jobs submitted.

Administrator
Manages the scheduling infrastructure.

Developer
Defines the jobs to be run specifying the job parameters, resource requirements, and so on.

Submitter
Manages the submission of their own jobs and monitors and controls the job lifecycle. This is the typical role for a IBM Workload Scheduler user.

IBM Workload Scheduler acts as submitter of jobs to the IBM Workload Scheduler dynamic agent.

Configurator
Is the entity responsible for running the jobs on a local environment.

To map security roles to users and groups on the WebSphere Application Server you must modify the **BrokerSecurityProps.properties** file using the **changeBrokerSecurityProperties** script.

To avoid the risk of changing a configuration value inadvertently or of overwriting the latest changes, you should always first create a file containing the current properties, edit it to the values you require, and apply the changes. Proceed as follows:

1. Log on to the computer where IBM Workload Scheduler is installed as the following user:

   UNIX root

   Windows
   Any user in the *Administrators* group.
2. Access the directory: *<TWA_home>*/wastools
3. Stop the WebSphere Application Server using the **conman stopappserver** command (see "Starting and stopping the application server and **appservman**" on page 450)
4. From that same directory run the following script to create a file containing the current broker security properties:

   UNIX **showBrokerSecurityProperties.sh > my_file_name**

   Windows
   showBrokerSecurityProperties.bat > my_file_name
5. Edit my_file_name with a text editor.
6. Edit the properties as you require. For each of the roles in the file, you can set the following properties:

   Everyone?
   Possible values:

   - **Yes**: Every user is authorized to perform tasks for the role. No check is performed on the WebSphere Application Server user registry.

- **No** : Access is denied to users not defined in the WebSphere Application Server user registry.

All authenticated?
Possible values:

- **Yes**: All users belonging to the current WebSphere Application Server user registry who have been authenticated can access resources and tasks for the role. This is the default value.
- **No** : Access is granted only to those users and groups defined in the WebSphere Application Server user registry and listed in the mapped user and mapped group properties.

Mapped users
If specified, one or more users separated by the vertical bar symbol (|). This field can be left blank.

Mapped groups
If specified, one or more groups separated by the vertical bar symbol (|). This field can be left blank.

7. Save the file *my_file_name*.
8. Run the script:

   Windows
   changeBrokerSecurityProperties.bat my_file_name

   UNIX **changeBrokerSecurityProperties.sh my_file_name**

   where *my_file_name* is the *fully qualified path* of the file containing the new parameters.

   The properties are updated, according to the rules given in the descriptions of each property type.
9. Start the WebSphere Application Server using the **conman startappserver** command (see "Starting and stopping the application server and **appservman**" on page 450)
10. Check that the change has been implemented.

Note:

1. If the mapped user or group names contain blanks, the entire user or group list must be specified between double quotation marks ("). For example, if you want to add the users John Smith, MaryWhite and DavidC to the developer role, you specify them as follows:

   ```
   Role: Developer
   Everyone?: No
   All authenticated?: No
   Mapped users:"John Smith|MaryWhite|DavidC"
   Mapped groups:
   ```
2. In the file there is an additional default role named **WSClient** which you must leave as is.

Examples: To assign the Operator role to users Susanna and Ann belonging to the current WebSphere Application Server user registry:

```
Role: Operator
Everyone?: No
All authenticated?: No
Mapped users:Susanna|Ann
Mapped groups:
```

To assign the Administrator role to user Tom and the Developer role to the user group MyGroup defined in the current WebSphere Application Server user registry:

```
Role: Administrator
Everyone?: No
All authenticated?: No
Mapped users:Tom
Mapped groups:

Role: Developer
Everyone?: No
All authenticated?: No
Mapped users:
Mapped groups:MyGroup
```

BrokerSecurityProps.properties file

```
################################################################
# Broker Security Properties
################################################################


Role: WSClient
Everyone?: No
All authenticated?: Yes
Mapped users:
Mapped groups:

Role: Administrator
Everyone?: No
All authenticated?: Yes
Mapped users:
Mapped groups:

Role: Operator
Everyone?: No
All authenticated?: Yes
Mapped users:
Mapped groups:

Role: Submitter
Everyone?: No
All authenticated?: Yes
Mapped users:
Mapped groups:

Role: Configurator
Everyone?: No
All authenticated?: Yes
Mapped users:
Mapped groups:

Role: Developer
Everyone?: No
All authenticated?: Yes
Mapped users:
Mapped groups:
```

Configuring command-line client access authentication

This section describes how to reconfigure the connection used by the command line client.

The command line client is installed automatically on the master domain manager and can be installed optionally on any other workstation. On the master domain manager you use it to run all of the commands and utilities.

On any other workstation you use it to run one of the following commands:

- **evtdef**
- **composer**
- **optman**
- **planman**
- **sendevent**

It is configured automatically by the installation wizard, but if you need to change the credentials that give access to the server on the master domain manager, or you want to use it to access a different master domain manager, modify the ***connection parameters*** as described here.

Note:

1. The ***connection parameters*** are not required to use the local **conman** program on a fault-tolerant agent.
2. The command-line client on the master domain manager uses exactly the same mechanism to communicate with the server as it does when it is installed remotely.

Connection parameters

About this task

The connection parameters can be provided in one of three ways:

Define them in localopts

All fields except *username* and *password*, can be defined by editing the *TWA_home*/TWS/**localopts** properties file on the computer from which the access is required. See "Setting local options" on page 34 for a full description of the file and the properties.

In **localopts** there is a section for the general connection properties, which contains the following:

```
host = host_name
protocol = protocol
port = port number
proxy = proxy server
proxyport = proxy server port number
timeout = seconds
defaultws = master_workstation
useropts = useropts_file
```

In addition, there are separate groups of SSL parameters which differ depending on whether your network is FIPS-compliant, and thus uses GSKit for SSL, or is not, and uses OpenSSL (see "FIPS compliance" on page 339 for more details):

FIPS-compliant (GSKit)

```
CLI SSL keystore file = keystore_file_name
CLI SSL certificate keystore label = label
CLI SSL keystore pwd = password_file_name
```

Not FIPS-compliant (OpenSSL)

```
CLI SSL server auth = yes|no
CLI SSL cipher = cipher_class
CLI SSL server certificate =certificate_file_name
CLI SSL trusted dir =trusted_directory
```

Store some or all of them in useropts

As a minimum, the **username** and **password** parameters can be defined in the *user_home*/.TWS/**useropts** file for the user who needs to make the connection. Also, if you need to personalize for a user any of the properties normally found in the localopts file, add the properties to the useropts file. The values in the useropts file always take precedence over those in the localopts file. See "Setting user options" on page 55 for a full description of the file and the properties.

The minimum set of properties you would find in **useropts** is as follows:

```
username=user_ID
password=password
```

Supply them when you use the command

When you use any of the commands you can add one or more of the connection parameters to the command string. These parameters take precedence over the parameters in **localopts** and **useropts**. This allows you, for example, to keep the parameters in the **localopts** file and just get users to supply the **username** and **password** parameters when they use one of the commands, avoiding the necessity to store this data in the **useropts** file for each user..

The parameters can either be supplied fully or partially in a file, to which you refer in the command string, or typed directly as part of the command string. The full syntax is as follows:

```
[-file <parameter_file>
|
 [-host <host_name>]
 [-password <user_password>]
 [-port <port_number>]
 [-protocol {http|https}]
 [-proxy <proxy_name>]
 [-proxyport <proxy_port_number>]
 [-timeout <timeout>]
 [-username <username>]
```

-file <*parameter_file*>
: A file containing one or more of the connection parameters. Parameters in the file are superseded if the corresponding parameter is explicitly typed in the command.

-host <*host_name*>
: The host name or IP address of the master domain manager to which you want to connect.

-password <*user_password*>
: The password of the user supplied in the -username parameter.

-port <*port_number*>
: The listening port of the master domain manager to which you want to connect.

-protocol {http|https}
: Enter either http or https, depending on whether you want to make a secure connection.

-proxy <*proxy_name*>
The host name or IP address of the proxy server involved in the connection (if any).

-proxyport <*proxy_port_number*>
The listening port of the proxy server involved in the connection (if any).

-timeout <*timeout*>
The number of seconds the command line client is to wait to make the connection before giving a timeout error.

-username <*username*>
The user ID of the user making the connection.

Note: From the command line, neither the default workstation, nor the command line client SSL parameters can be supplied. These must always be supplied in either the `localopts` (see "Setting local options" on page 34) or the `useropts` file for the user (see "Setting user options" on page 55).

The command line client needs to assemble a full set of parameters, and it does so as follows:

1. First it looks for values supplied as parameters to the command
2. Then, for any parameters it still requires, it looks for parameters supplied in the file identified by the `-file` parameter
3. Then, for any parameters it still requires, it looks in the `useropts` file for the user
4. Finally, for any parameters it still requires, it looks in the `localopts` file

If a setting for a parameter is not specified in any of these places an error is displayed.

Entering passwords

About this task

Password security is handled as follows:

Password entered in `useropts` file
You type the connection password into the `useropts` file in unencrypted form. When you access the interface for the first time it is encrypted. This is the preferred method.

Password entered in the parameter file used by the command
You type the connection password into the parameter file in unencrypted form. It is not encrypted by using the command. Delete the file after use to ensure password security.

Password entered using the `-password` parameter in the command
You type the password in the command string in unencrypted form. It remains visible in the command window until you clear the command window contents.

Note: On Windows workstations, when you specify a password that contains double quotation marks (") or other special characters, make sure that the character is escaped. For example, if your password is `tws11"tws`, write it as `"tws11\"tws"` in **`useropts`**.

IBM Workload Scheduler console messages and prompts

The IBM Workload Scheduler control processes (Netman, Mailman, Batchman, Jobman, and Writer) write their status messages (referred to as console messages) to standard list files. These messages include the prompts used as job and Job Scheduler dependencies. On UNIX and Linux operating systems, the messages can also be directed to the **syslog** daemon (**syslogd**) and to a terminal running the IBM Workload Scheduler console manager. These features are described in the following sections.

Setting sysloglocal on UNIX

About this task

If you set **sysloglocal** in the local options file to a positive number, IBM Workload Scheduler's control processes send their console and prompt messages to the **syslog** daemon. Setting it to **-1** turns this feature off. If you set it to a positive number to enable system logging, you must also set the local option **stdlistwidth** to **0**, or a negative number.

IBM Workload Scheduler's console messages correspond to the following **syslog** levels:

LOG_ERR
Error messages such as control process abends and file system errors.

LOG_WARNING
Warning messages such as link errors and stuck job streams.

LOG_NOTICE
Special messages such as prompts and tellops.

LOG_INFO
Informative messages such as job launches and job and Job Scheduler state changes.

Setting **sysloglocal** to a positive number defines the syslog facility used by IBM Workload Scheduler. For example, specifying **4** tells IBM Workload Schedulerto use the local facility LOCAL4. After doing this, you must make the appropriate entries in the **/etc/syslog.conf** file, and reconfigure the syslog daemon. To use LOCAL4 and have the IBM Workload Scheduler messages sent to the system console, enter the following line in **/etc/syslog.conf**:

```
local4    /dev/console
```

To have the IBM Workload Scheduler error messages sent to the **maestro** and **root** users, enter the following command:

```
local4.err    maestro,root
```

The selector and action fields must be separated by at least one tab. After modifying **/etc/syslog.conf**, you can configure the **syslog** daemon by entering the following command:

```
kill -HUP `cat /etc/syslog.pid`
```

console command

About this task

You can use the conman **console** command to set the IBM Workload Scheduler message level and to direct the messages to your terminal. The message level setting affects only Batchman and Mailman messages, which are the most numerous. It also sets the level of messages written to the standard list file or files and the **syslog** daemon. The following command, for example, sets the level of Batchman and Mailman messages to **2** and sends the messages to your computer:

```
console sess;level=2
```

Messages are sent to your computer until you either run another **console** command, or exit conman. To stop sending messages to your terminal, enter the following conman command:

```
console sys
```

Enabling the time zone feature

About this task

Time zones are enabled by default on installation of the product.

When you upgrade, the time zone feature inherits the setting of the previous installation. You can enable the time zone using the **enTimeZone** option of the **optman** command, as follows:

```
optman chg enTimeZone = yes
```

The following steps outline the method of implementing the time zone feature:

1. Load IBM Workload Scheduler.

   The database allows time zones to be specified for workstations, but not on *start* and *deadline* times within job streams in the database. The plan creation (JnextPlan) ignores any time zones that are present in the database. You will not be able to specify time zones anywhere in the plan.
2. Define workstation time zones.

   Set the time zone of the master domain manager workstation, of the backup master domain manager, and of any agents that are in a different time zone than the master domain manager. No time zones are allowed in the database for **Start**, **Latest Start Time**, and **Termination Deadline** times. No time zones are allowed anywhere in the plan at this point, because **enTimeZone** is set to **no**.
3. When workstation time zones have been set correctly, enable the time zone feature.

   All users are able to use time zones anywhere in the database, although they should wait for the next run of JnextPlan to use them on **Start**, **Latest Start Time**, and **Termination Deadline** times. The next time JnextPlan runs, time zones are carried over to the plan and the Dynamic Workload Console, and the back end allows specification of time zones anywhere in the plan.
4. Start using time zones on *start* and *until* times where needed.

   You can now use all time zone references in the database and in the plan with the Dynamic Workload Console and the command-line interface.

Configuring to use the report commands

You use the IBM Workload Scheduler report commands to obtain summary or detailed information about your workload scheduling. Before using these commands, however, they must be configured for your environment. This process is described in the chapter on getting reports and statistics in the *IBM Workload Scheduler: User's Guide and Reference*.

Modifying jobmon service rights for Windows

On Windows systems, the IBM Workload Scheduler jobmon service runs in the SYSTEM account with the right **Allow Service to Interact with Desktop** granted to it. You can remove this right for security reasons. However, if you do so, it prevents the service from launching interactive jobs that run in a window on the user's desktop. These jobs will be run, but are not accessible from the desktop or from IBM Workload Scheduler and do not have access to desktop resources. As a result, they may run forever or abend due to lack of resources.

Chapter 3. Configuring the Dynamic Workload Console

This chapter describes how to configure Dynamic Workload Console. It is divided into the following sections:

Personalizing UI labels on the Dynamic Workload Console

Dynamic Workload Console provides the capability to customize user interface labels.

Before you begin

You might find this feature useful for your business users so that the tasks they perform are in the context of your line of business. You can personalize the UI labels for the Dynamic Workload Console, for example, you can rename the **Job Stream View** by clicking on the **Rename** icon in the upper left corner. The **Rename** icon is displayed when you hover over the Job Stream View string.

For information about customizing user interface labels on the Dynamic Workload Console, see Console identity string in Administering applications and their environment.

Launching in context with the Dynamic Workload Console

Create a URL to launch the Dynamic Workload Console and have it directly open the results of a particular query.

You can then include this URL in an external application, for example, to monitor jobs and job streams that are critical to your business, and to quickly and easily manage them. You can access specific job or job stream details without having to create customized queries and monitor the state and health of the workstations that are critical in your environment so that, when unavailability or malfunctioning impacts job scheduling, you are alerted.

Scenarios

The following main scenarios can be identified:

- Obtain the result of a Monitor task on:
  - Jobs
  - Critical jobs
  - Job streams
- Obtain the result of a Monitor task on workstations
- Obtain the result of a saved task.

For all the scenarios, you must create a basic URL as described in "Creating a basic URL."

Creating a basic URL
About this task

To create a basic URL, perform the following steps:

Procedure

1. Define the URL to access the Dynamic Workload Console:

   ```
   https://{WebUIHostname:adminSecurePort}
   /DASH_context_root/xLaunch.do?pageID=com.ibm.tws.
   WebUI.External.navigation&showNavArea=false
   ```

   where:

   WebUIHostname
   It is the fully qualified hostname or the IP address of the computer where the Dynamic Workload Console is installed.

   adminSecurePort
   It is the number of the port on which the Dynamic Workload Console is listening.

   DASH_context_root
   It is the Dashboard Application Services Hub context root defined at installation time. The context root determines the URL of a deployed application and by default is identical with the application directory or archive structure. In this case, the default is `ibm/console`.

   Example

   ```
   https://mypc:29443/ibm/console/xLaunch.do?pageID=com.ibm.tws.WebUI.
   External.navigation&showNavArea=false
   ```

2. Specify the action that you want to run, by specifying the corresponding parameter:

   &action
   It indicates the action that you want to perform and can have one of the following values:
   - BrowseJobs
   - ZBrowseJobs
   - BrowseJobStreams
   - BrowseCriticalJobs
   - BrowseWorkstation
   - InternalTask

3. Specify the engine on which you want to run the query, by entering its parameters:

   &hostname
   For distributed environments, it is the host name or TCP/IP address of the computer on which the IBM Workload Scheduler engine is installed. For z/OS® environments, it is the host name or TCP/IP address of the computer on which the z/OS connector is installed.

   &port The port number that is used to connect to the computer on which the IBM Workload Scheduler engine or the z/OS connector is installed. Typically, the default port numbers are:

Table 26. Default port numbers

Port number	Engine
31117	IBM Workload Scheduler distributed engine
31127	IBM Workload Scheduler for z/OS engine with z/OS connector V.8.3
31217	IBM Workload Scheduler for z/OS engine with z/OS connector V.8.5 or higher

Table 26. Default port numbers (continued)

Port number	Engine
2809	IBM Workload Scheduler for z/OS with z/OS connector on z/OS WebSphere Application Server
16312	IBM Workload Scheduler for z/OS engine V. 9.1

&server
It applies to z/OS systems only and is mandatory. It is the name of the remote server of the engine as it was specified in the z/OS connector.

Example

```
&hostname = webuidev&port = 31217&server = C851
```

Results

Example of a complete URL:

```
https://mypc:29443/ibm/console/xLaunch.do?pageID=
com.ibm.tws.WebUI.External.navigation&showNavArea=false
&action=BrowseJobs&hostname=webuidev&port=31117
```

Creating a URL to launch the Plan View in context

About this task

To create a URL to launch the Plan View in context, perform the following steps:

Procedure

1. Define the URL as follows:

   ```
   https://<WebUIHostname>:<WebUI_port>/dwc/
   faces/html/jsp/PlanViewerLauncher.jsp?engineName=<ENGINE_NAME>
   ```

   where:

   WebUIHostname
   It is the fully qualified hostname or the IP address of the computer where the Dynamic Workload Console is installed.

   WebUI_port
   It is the port used to access the Dynamic Workload Console.

   engineName
   It is the name of the engine defined in the Dynamic Workload Console.

   Example

   ```
   https://myhost.softwarelab.it.mycompany.com:16311/dwc/faces/html
   /jsp/PlanViewerLauncher.jsp?engineName=ROME
   ```

2. You can also add some optional parameters at the end of the URL string with the `&<param1>=<value>&param2=value` syntax. The optional parameters are as follows.

   jsNameFilter
   It is the name of the job streams to filter. It can contain wildcard characters (*).

   wksNameFilter
   It is the name of the workstations to filter. It can contain wildcard characters (*)

predecessors
Set to `true` if you want to include the predecessors of the selected job stream. This setting has no effect is no if no filter is specified.

successor
Set to `true` if you want to include the successors of the selected job stream. This setting has no effect is no if no filter is specified.

arrivalTimeStart
When specifying the Scheduled Time range, this value indicates the **from** date. It must be in the "dd.MM.yy.hh.mm.ss" form, where: dd is the day, MM is the month, yy is the year, hh is the hour, mm are the minutes, ss are the seconds

arrivalTimeEnd
When specifying the Scheduled Time range, this value indicates the **to** date. It must be in the "dd.MM.yy.hh.mm.ss" form, where: dd is the day, MM is the month, yy is the year, hh is the hour, mm are the minutes, ss are the seconds

Results

Example of a complete URL:

```
https://myhost.romelab.it.ibm.com:16311/dwc/faces/html/jsp
/PlanViewerLauncher.jsp?engineName=ROME&engineOwner=admin&jsNameFilter=j*
```

Advanced optional parameters

Depending on the query whose results you want to view, you can complete your URL with the following parameters.

Monitor Jobs on distributed systems

Create a URL by specifying the **BrowseJobs** action, as described in "Creating a basic URL" on page 101.

You can also specify any of the following filters:

&workstation
Filter by the workstation on which the jobs runs.

&jobstream
Filter by the job stream that contains the jobs.

&job Filter by the job name.

&schedtime
Filter by the job scheduled time.

&status
Filter by the job status. You can filter by one or more statuses. Possible values are:

W Waiting
O Successful
H Held
R Ready
E Error
U Undecided
S Running
C Canceled
B Blocked

&columns
Specify the number of columns that you want to display in your result table. If not specified, the default number of columns for this query is shown. Supported values are:
Min Display a minimum set of columns
All Display all columns

Example:

```
https://mypc:29043/DASH_context_root/xLaunch.do?pageID=com.ibm.tws.WebUI.
External.navigation&showNavArea=false&action=BrowseJobs
&hostname=webuidev&port=31117
&workstation=my_ws&jobstream=my_js_name&job=my_job_name&status=ESB&columns=ALL
```

where:

DASH_context_root
It is the Dashboard Application Services Hub context root defined at installation time. The context root determines the URL of a deployed application and by default is identical with the application directory or archive structure. In this case, the default is `ibm/console`.

Monitor Jobs on z/OS systems

Create a URL by specifying the **ZBrowseJobs** action, as described in "Creating a basic URL" on page 101.

You can also specify any of the following filters:

&workstation
Filter by the workstation on which the job runs.

&jobstream
Filter by the job stream that contains the jobs.

&job Filter by the job name.

&schedtime
Filter by the job scheduled time.

&columns
Specify the number of columns that you want to display in your result table. If not specified, the default number of columns for this query is shown. Supported values are:

Min display a minimum set of columns

All display all columns

Example:

```
https://mypc:29043/DASH_context_root/xLaunch.do?pageID=com.ibm.tws.WebUI.
External.navigation&showNavArea=false&action=ZBrowseJobs
&hostname=webuidev&port=31117
&server=C851&workstation=my_ws&jobstream=my_js_name
&job=my_job_name&schedtime=200812081100
```

where:

DASH_context_root
It is the Dashboard Application Services Hub context root defined at installation time. The context root determines the URL of a deployed application and by default is identical with the application directory or archive structure. In this case, the default is `ibm/console`.

Monitor Critical Jobs

Create a URL by specifying the **BrowseCriticalJobs** action, as described in "Creating a basic URL" on page 101.

You can also specify any of the following filters:

&workstation
Filter by the workstation on which the job runs.

&jobstream
Filter by the job stream that contains the jobs.

&job Filter by the job name.

&schedtime
Filter by the job scheduled time.

&columns
Specify the number of columns that you want to display in your result table. If not specified, the default number of columns for this query is shown. Supported values are:

Min Display a minimum set of columns

All Display all columns

Example:

```
https://mypc:29043/DASH_context_root/xLaunch.do?pageID=com.ibm.tws.WebUI.
External.navigation&showNavArea=false&action=BrowseCriticalJobs
&hostname=webuidev&port=31117
&workstation=my_ws&jobstream=my_js_name
&job=my_job_name&server=C851&columns=Min
```

where

&server
is a parameter used for z/OS only.

DASH_context_root
It is the Dashboard Application Services Hub context root defined at installation time. The context root determines the URL of a deployed application and by default is identical with the application directory or archive structure. In this case, the default is `ibm/console`.

Monitor Job Streams

Create a URL by specifying the **BrowseJobStreams** action, as described in "Creating a basic URL" on page 101.

You can also specify any of the following filters:

&workstation
Valid for distributed system only. Filter by the workstation on which the job stream runs.

&jobstream
Filter by the job stream name.

&columns
Specify the number of columns that you want to display in your result table. If not specified, the default number of columns for this query is shown. Supported values are:

Min Display a minimum set of columns

All Display all columns

Example:

```
https://mypc:29043/DASH_context_root/xLaunch.do?pageID=com.ibm.tws.WebUI.
External.navigation&showNavArea=false&action=BrowseJobStreams
&hostname=webuidev&port=31117
&workstation=my_ws&jobstream=my_js_name
&server=C851&columns=ALL
```

where,

DASH_context_root
It is the Dashboard Application Services Hub context root defined at installation time. The context root determines the URL of a deployed application and by default is identical with the application directory or archive structure. In this case, the default is `ibm/console`.

server Parameter used for z/OS systems only.

Monitor Workstations

Create a URL specifying the **BrowseWorkstation** action, as described in "Creating a basic URL" on page 101.

You can also specify any of the following filters:

&workstation
Filter by the workstation name.

&columns
Specify the number of columns that you want to display in your result table. If not specified, the default number of columns for this query is shown. Supported values are:

Min Display a minimum set of columns

All Display all columns

Example:

```
https://mypc:29043/DASH_context_root/xLaunch.do?pageID=com.ibm.tws.WebUI.
External.navigation&showNavArea=false&action=BrowseWorkstation
&hostname=webuidev&port=31117
&workstation=my_ws&jobstream=my_js_name
&server=C851&columns=ALL
```

where:

&server
is a parameter used for z/OS only.

DASH_context_root
It is the Dashboard Application Services Hub context root defined at installation time. The context root determines the URL of a deployed application and by default is identical with the application directory or archive structure. In this case, the default is `ibm/console`.

Existing task

About this task

Create a URL by specifying the **InternalTask** action, as described in "Creating a basic URL" on page 101.

You can save this URL can be saved as a bookmark in your browser, so that, by clicking the bookmark, you can directly open the results of a previously created task.

To save a task URL, perform the following steps:

1. Create a task with the Dynamic Workload Console:

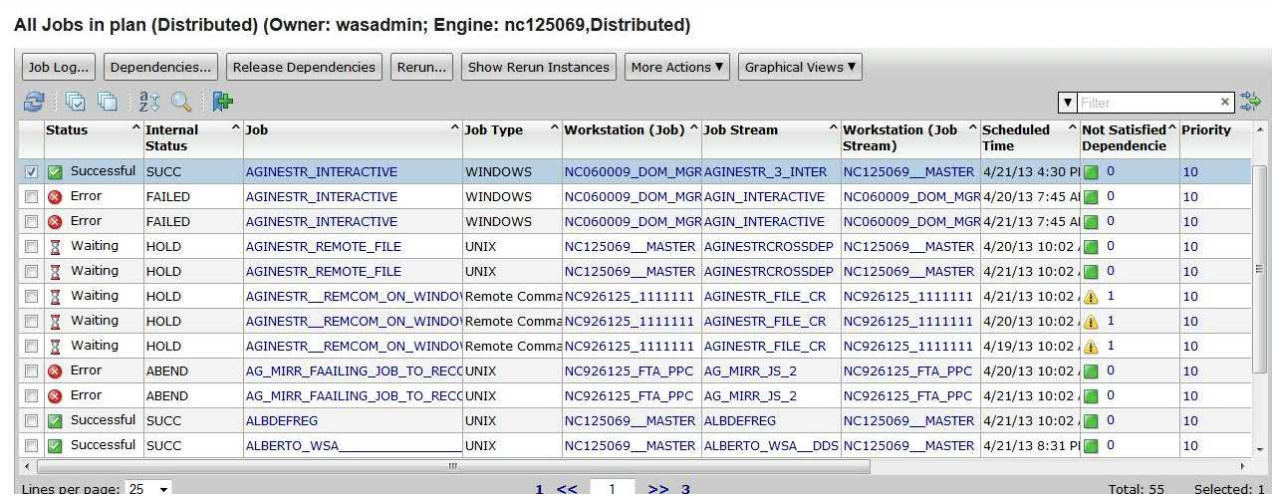


Figure 1. List of tasks

2. From the displayed panel, click the **Add bookmark icon**

   to save this link in your bookmarks.
3. Specify a name for the new bookmark. By default the task name is used. Organize your bookmarks for your convenience, for example, you might organize your saved tasks in a different folder for each engine.

Example of a saved bookmark:

```
https://cairapc:29043/ibm/console/xLaunch.do?pageID=com.ibm.tws.WebUI.
External.navigation&showNavArea=false
&action=InternalTask&hostname=fferrar4&port=31117
&taskname=All%20Jobs%20in%20plan%20for%20ferrari
```

Starting from this bookmark you can manually create a URL as follows:

```
https://mypc:29043/DASH_context_root/xLaunch.do?pageID=com.ibm.tws.WebUI.
External.navigation&showNavArea=false&action= InternalTask
&hostname=webuidev&port=31117
&server=C851 &taskname=myTask
```

where:

&server
: is a parameter used for z/OS only.

DASH_context_root
: It is the Dashboard Application Services Hub context root defined at installation time. The context root determines the URL of a deployed application and by default is identical with the application directory or archive structure. In this case, the default is `ibm/console`.

Configuring access to the Dynamic Workload Console

As soon as you finish installing the Dynamic Workload Console, you can launch it by using the link provided in the final installation panel.

However, after the installation, the administrator is the only user who can log into the console, using the credentials specified during the installation.

This is the user defined in the WebSphere Application Server file registry (WIM).

There are two important steps an administrator must perform before other users can log into and use the Dynamic Workload Console

- "Configuring a user registry" on page 110
- "Configuring roles to access the Dynamic Workload Console" on page 111

If WebSphere Application Server is configured to use the file registry (WIM) user registry (the default value), users and groups must be created in the file registry using the WebSphere Application Server administrative console:

To create and assign roles, log in to Dashboard Application Services Hub and run the following steps:

1. From the navigation toolbar, click the search glass icon, on top of the toolbar. In the search field, enter WebSphere Administrative Console to open the administrative console.
2. Click **Launch WebSphere administrative console**.
3. From the administrative console navigation tree, click **Users and Groups** > **Manage users** to create a new user on the file registry (do not create it on the operative system).

For more information about creating and assigning roles, see the Dashboard Application Services Hub online help by clicking the "**?**" (question mark) in top-right corner of the panels.

If WebSphere Application Server is configured to use the local operating system user registry, users and groups must be created in the local operating system. However, you can replace the local operating system user registry with LDAP or a file registry and vice versa, or configure the use of more than one of these.

Users defined in the user registry can log in to the Dynamic Workload Console; then they need to be associated to a role to be able access the Dynamic Workload Console features (see ""Configuring roles to access the Dynamic Workload Console" on page 111.)

By default, the Dynamic Workload Console uses file registry (WIM) for authentication purposes. If you want to switch to local OS or PAM authentication, perform the steps described in "Configuring the Dynamic Workload Console to use the local OS or PAM authentication method"

If the Dynamic Workload Console on UNIX operating systems uses PAM (Pluggable Authentication Module) for authentication purposes and you want to switch to local OS authentication, perform the steps defined in "Configuring the Dynamic Workload Console to use the local OS or PAM authentication method."

Note: If two or more instances of Dynamic Workload Console share the same database repository for their settings, but they are not configured to be in a High Availability configuration, they all must be at the same fix pack level.

Configuring a user registry

If WebSphere Application Server is configured to use LDAP user registry, users and groups must be created by the system administrator in the chosen LDAP server database.

Configuring user registries for the Dynamic Workload Console and all other IBM Workload Scheduler components is described in Chapter 5, "Configuring authentication," on page 245.

Configuring the Dynamic Workload Console to use the local OS or PAM authentication method

About this task

To modify the Dynamic Workload Console authentication method to use the local OS or PAM authentication method, perform the following steps:

Procedure

1. Log in to the Dynamic Workload Console with WebSphere Application Server administration credentials.
2. From the navigation toolbar, click the search glass icon, on top of the toolbar. In the search field, enter WebSphere Administrative Console to open the administrative console.
3. Click **Launch WebSphere administrative console**.
4. From the administrative console navigation tree, click **Users and Groups** > **Manage users** to create a new user on the file registry (do not create it on the operating system).
5. Switch back to the Dynamic Workload Console.
6. From the navigation toolbar, click the search glass icon, on top of the toolbar. In the search field, enter `User Roles` to open the **User Roles** page.
7. Enter the new user account in the **User ID** field and click **Search**.
8. Click on the user name in the results table.
9. Select the roles the for primary administrative user. Usually all roles are assigned.

10. Press the **Save** button.
11. Backup the WebSphere Application Server configuration using the **backupConfig** command.
12. Dump your current security properties to a text file by using the following command:

    ```
    showSecurityProperties.sh > text_file
    ```

13. Customize the security properties by editing the file as follows:

    Note: If you want to use PAM authentication, specify the following property in the file `activeUserRegistry=Custom`.

    ```
    ###############################################################
    Global Security Panel
    ###############################################################
    enabled=true
    enforceJava2Security=false
    useDomainQualifiedUserNames=false
    cacheTimeout=600
    ltpaTimeOut=720
    issuePermissionWarning=false
    activeProtocol=CSI
    useFIPS=false
    activeAuthMechanism=LTPA
    activeUserRegistry=LocalOS

    ###############################################################
    Federated Repository Panel
    ###############################################################
    PrimaryAdminId=new_user
    UseRegistryServerId=true
    ServerID=new_user
    ServerPassword=new_pwd
    VMMRealm=TWSREALM
    VMMRealmDelimiter=@
    VMMIgnoreCase=true
    ```

14. Stop the server by using the **stopWas.sh** wastool. To stop the server, use the WebSphere Application Server administration credentials.
15. Load the new properties by entering the following command:

    ```
    complete_path/changeSecurityProperties.sh text_file
    ```

16. Restart the server by using the **startWas.sh** wastool.

Configuring roles to access the Dynamic Workload Console

During the Dynamic Workload Console installation, new predefined roles are created in the Dashboard Application Services Hub. They determine which console panels are available to a user, and therefore which activities that user can perform from Dynamic Workload Console.

If you do not assign any of the predefined roles to a Dashboard Application Services Hub user, that user, after having logged in, will not see any entry for IBM Workload Scheduler, and dynamic workload broker in the navigation tree.

Note: The users will not be able to view the console panels if the required engines are not shared with them.

Depending on the security repository you use, the following points apply:

LocalOS
: Create the user in the operating system and assign the roles to that user using the Dashboard Application Services Hub console.

WIM Create the user using the WebSphere administrative console. The user is a WebSphere Application Server user. Then assign roles by using the Dashboard Application Services Hub console as follows: in the Dashboard Application Services Hub navigation bar, click the settings icon that opens the **Console Settings** menu and click **Roles**.

To create and assign roles, log in to Dashboard Application Services Hub and run the following steps:

1. From the navigation toolbar, click the search glass icon, on top of the toolbar. In the search field, enter WebSphere Administrative Console to open the administrative console.
2. Click **Launch WebSphere administrative console**.
3. From the administrative console navigation tree, click **Users and Groups** > **Manage users** to create a new user on the file registry (do not create it on the operative system).

For more information about creating and assigning roles, see the Dashboard Application Services Hub online help by clicking the "**?**" (question mark) in top-right corner of the panels.

Tip It is not necessary to assign a role to every single user. If the user registry already contains groups of users that are properly defined for using the console, it is possible to assign roles to groups too. If groups are not available in the user registry, then the special role **all authenticated portal users** can be used to assign roles to *all* the users at once.

Within Dashboard Application Services Hub, you can create your own custom views to enable users to see all or a subset of IBM Workload Scheduler pages. To do it, you must have the **wasadmin** role and perform the following steps:

1. Create a new Dashboard Application Services Hub View with the pages you want to be available:
   a. In the Dashboard Application Services Hub navigation bar, click the settings icon that opens the Console Settings menu and click **Views**.
   b. Click **New**.
   c. In the Views panel, click **New** and specify a name for the new view.
   d. In the Views page, Expand **Pages in This View** and click **Add** to add pages to the new view.
   e. Select the pages you want to be available to this view and click **Add**
2. Add the created view to the **all authenticated portal users** role:
   a. In the Dashboard Application Services Hub navigation bar, click the settings icon that opens the Console Settings menu and click **Roles**.
   b. Click **all authenticated portal users** role.
   c. Expand **Access to Views** section, click the **Add** (plus sign) icon, select the newly created view that you want to add to this role and click **Add**.
   d. Expand **Access to Views** and select the pages to which the users of this role must have access.

The following lists the predefined roles created in the Dashboard Application Services Hub for accessing the IBM Workload Scheduler environments using Dynamic Workload Console:

TWSWEBUIAdministrator
Users in this group can see the entire portfolio and use all features of the Dynamic Workload Console.

Users in this group can also access and use all features of the Self-Service Catalog and theSelf-Service Dashboards mobile applications. From the Self-Service Catalog mobile application, these users can create and edit catalogs, create and edit services, add services to catalogs, submit services associated to job streams, and share catalogs and services with other users. From the Self-Service Dashboards mobile application, these users can create and edit dashboards to filter for jobs and workstations, display a dashboard of results, perform recovery actions on a single result.

TWSWEBUIConfigurator
Users in this group can manage Dynamic Workload Console scheduler connections, user preferences, and scheduling environment design.

TWSWEBUIOperator
Users in this group can see Dynamic Workload Console:

- All Monitor tasks
- Jobs and job streams to be submitted on request
- Set User Preferences

TWSWEBUIDeveloper
Users in this group can create, list, and edit workload definitions, workstations, and event rule definitions in the IBM Workload Scheduler database.

TWSWEBUIAnalyst
Users in this group can manage Dynamic Workload Console reports and user preferences.

Users in this group can also access the Self-Service Catalog and theSelf-Service Dashboards mobile applications but the actions they can perform are limited to submitting service requests (job streams) from the Self-Service Catalog and, from the Self-Service Dashboards mobile application, displaying a dashboard of results and performing recovery actions on them.

TWSWEBUIBusinessDeveloper
Users in this group can access and use the Self-Service Catalog and the Self-Service Dashboards mobile applications. From the Self-Service Catalog mobile application, these users can create and edit catalogs, create and edit services, add services to catalogs, delete services and catalogs, and submit services associated to job streams. From the Self-Service Dashboards mobile application, these users can create and edit dashboards to filter for jobs and workstations, display and view a dashboard of results, delete dashboards, and perform recovery actions on a single result. To share catalogs, services, and dashboards with other users, the TWSWEBUIBusinessDeveloper can assign them to the custom roles that the TWSWEBUIBusinessDeveloper possesses but not to predefined roles. Users with these same custom roles can work with the catalogs, services, and dashboards. Users with all of the custom roles can submit services; view, edit, and delete services, catalogs, and dashboards; but users with only one or some of the custom roles can only submit services, and view services, catalogs, and dashboards.

If a user with the Administrator role, creates catalogs, services, and dashboards and does not assign any roles to them, then users with the TWSWEBUIBusinessDeveloper role cannot see or work with them.

Note: If a custom role is removed from a catalog, service, or dashboards, in addition to the TWSWEBUIBusinessDeveloper user, users with this same custom role can no longer see and work with them, even if they possess

other custom roles that are currently assigned to the catalog or service. The Administrator must reassign the custom role to the catalog, service, or dashboard to make it accessible again to the TWSWEBUIBusinessDeveloper user and other users with the same custom role.

The following table lists some entries of the navigation toolbar, and some activities that you can perform on the Dynamic Workload Console. Beside each item, the table shows the groups whose users are authorized to access them.

Table 27. Menu and Group Permissions

Menu Item	**Groups with Permission**
Quick Start	TWSWEBUIAdministrator
All Configured Tasks	TWSWEBUIAdministrator TWSWEBUIOperator
Manage Workload Reports	TWSWEBUIAdministrator TWSWEBUIAnalyst
Administration -> Workload Design	TWSWEBUIAdministrator TWSWEBUIDeveloper
Administration -> Workload Forecast	TWSWEBUIAdministrator TWSWEBUIOperator
Administration -> Workload Submission	TWSWEBUIAdministrator TWSWEBUIOperator
Administration -> Monitor	TWSWEBUIAdministrator TWSWEBUIOperator
Administration -> Workload Design	TWSWEBUIAdministrator TWSWEBUIConfigurator
Administration -> Monitor	TWSWEBUIAdministrator TWSWEBUIOperator
Reporting	TWSWEBUIAdministrator TWSWEBUIAnalyst
System Configuration ->Manage Engines	TWSWEBUIAdministrator TWSWEBUIConfigurator
System Configuration -> Set User Preferences	TWSWEBUIAdministrator TWSWEBUIOperator TWSWEBUIConfigurator TWSWEBUIDeveloper TWSWEBUIAnalyst
System Configuration -> Manage Settings	TWSWEBUIAdministrator

Assigning a predefined role to an Dashboard Application Services Hub user allows that user to access the Dynamic Workload Console panels. The IBM Workload Scheduler user specified in the engine connection, instead, determines which operations can be run locally on the connected IBM Workload Scheduler engine. For example, if the user specified in a IBM Workload Scheduler engine connection is not authorized to run reporting in the IBM Workload Scheduler *Security file*, then, even though the Dashboard Application Services Hub user logged in to Dynamic Workload Console can access the reporting panels, he or she cannot perform reporting operations on that specific IBM Workload Scheduler engine. For more information about how to configure the security file, see "Configuring the security file" on page 208

The following lists the predefined roles created in the Dashboard Application Services Hub for accessing the dynamic workload broker environments using Dynamic Workload Console, and the panels they can access:

TDWBAdministrator
All panels

TDWBOperator
Scheduling Environment
Configuration
Definitions, except Define a New Job
Monitoring
Preferences

TDWBDeveloper
Configuration
Definitions
Preferences

TDWBConfigurator
Scheduling Environment
Configuration
Monitoring, except Job Instances
Preferences

Configuring WebSphere to authenticate the local OS or domain user

Configuring WebSphere to authenticate the local OS or domain user

About this task

If after installing the Dynamic Workload Console and enabling the Local OS security, you can not create an engine connection or if the configured task list is no longer displayed, perform the following steps:

Procedure

1. Stop the Dynamic Workload Console
2. Open the file in: <JAZZSM_HOME/profile/config/cells/JazzSMNode01Cell/wim/config/wimconfig.xml
3. At the end of the list of configuration properties:

```
<config:repositories adapterClassName="com.ibm.ws.wim.adapter.urbridge.URBridge"
  id="twaLocalOS" supportPaging="false">
 <config:baseEntries name="o=twaLocalOS"/>
 <config:CustomProperties name="uniqueUserIdProperty" value="uniqueId"/>
 <config:CustomProperties name="userSecurityNameProperty" value="uniqueName"/>
 <config:CustomProperties name="userDisplayNameproperty" value="displayName"/>
 <config:CustomProperties name="uniqueGroupIdProperty" value="uniqueId"/>
 <config:CustomProperties name="groupDisplayNameProperty" value="displayName"/>
 <config:CustomProperties name="groupSecurityNameProperty" value="uniqueName"/>
</config:repositories>
```

   add the following rows:

```
<config:CustomProperties name="com.ibm.websphere.registry.UseRegistry" value="local"/>
```

   for Domain User:

```
<config:CustomProperties name="com.ibm.websphere.registry.UseRegistry" value="domain"/>
```

If setting `local`, WebSphere will authenticate OS local user only. If setting `domain`, WebSphere will only authenticate the domain account from the Domain Controller.

4. Restart the Dynamic Workload Console and either the list of configured tasks or the created engine connection is displayed. If it is needed to authenticate both users, perform the following steps:
   a. Open the file <JAZZSM_HOME/profile/config/cells/JazzSMNode01Cell/wim/config/wimconfig.xml
   b. set the following property: <config:CustomProperties name="localAndDomainRegistryReturnPrincipalNameAsUserId" value="userId"/>

Configuring Dynamic Workload Console to use Single Sign-On

Single Sign-On (SSO) is a method of access control that allows a user to authenticate once and gain access to the resources of multiple applications sharing the same user registry.

This means that using SSO you can run queries on the plan or manage object definitions on the database accessing the engine without authenticating, automatically using the same credentials you used to log in to the Dynamic Workload Console.

The same is true when working with the Self-Service Catalog and Self-Service Dashboards apps from a mobile device. If the Dynamic Workload Console has been configured to use SSO, then these apps automatically use the same credentials used to log in to the Dynamic Workload Console.

After the installation completes you can configure Dynamic Workload Console and the IBM Workload Scheduler engine to use SSO. To do this they must share the same LDAP user registry.

The Lightweight Directory Access Protocol (LDAP) is an application protocol for querying and modifying directory services running over TCP/IP - see Chapter 5, “Configuring authentication,” on page 245 for more details.

If you configured Dynamic Workload Console to use Single Sign-On with an engine, then, the following behavior is applied:

If engine connection has the user credentials specified in its definitions
These credentials are used. This behavior regards also engine connections that are shared along with their user credentials.

If the user credentials are not specified in the engine connection
The credentials you specified when logging in to Dynamic Workload Console are used. This behavior regards also shared engine connections having unshared user credentials.

Before you proceed, ensure that the same value is defined for the `WMMRealm` property in both the Dynamic Workload Console and master domain manager. For more information about how to verify and correct this setting, see “Configuring the Dynamic Workload Console and master domain manager for Single Sign On” on page 117.

In addition to sharing the same LDAP user registry, the instance of WebSphere Application Server used by the Dynamic Workload Console and also the instance

used by the engine where the Single Sign-On is required, must both be configured to use the same Lightweight Third-Party Authentication token-keys. See "Configuring the use of Lightweight Third-Party Authentication" on page 118

Configuring the Dynamic Workload Console and master domain manager for Single Sign On

Configure the Dynamic Workload Console and master domain manager to enable Single Sing-On

About this task

Enabling Single Sing-On between Dynamic Workload Console and master domain manager requires that the `WMMRealm` property in your security settings be set to the same value in both the Dynamic Workload Console and master domain manager. The value itself is not relevant, but it must be same for both components. The default value for this property is **WIMDefaultFileBasedRealm** on the Dynamic Workload Console and **TWSREALM** on the master domain manager.

To enable Single Sing-On between the Dynamic Workload Console and master domain manager, perform the following steps:

Procedure

1. Check the value of the `WMMRealm` property on both the Dynamic Workload Console and master domain manager, as follows:
   a. Browse to the folder where the WebSphere Application Server tools (or wastools) are stored:

      On the Dynamic Workload Console
      <TDWC_INSTALL_PATH>/wastools

      On the master domain manager
      `<TWA_home>/wastools`

   b. Run the following command:

      UNIX **`showSecurityProperties.sh`**

      Windows
      `showSecurityProperties.bat`

   c. Check that the `WMMRealm` property is set to the same value on both the Dynamic Workload Console and master domain manager. If the two values are equal, no other actions are required. If the two values are different, proceed to step 2
2. You can perform this series of steps on either the Dynamic Workload Console or the master domain manager indifferently. Backup the WebSphere Application Server configuration using the following command:

   UNIX **`backupConfig.sh`**

   Windows
   `backupConfig.bat`
3. Dump your current security properties to a text file by using the following command:

   UNIX **`showSecurityProperties.sh`** > *text_file*

   Windows
   `showSecurityProperties.bat` > *text_file*

4. Customize the security properties by setting the value for the `WMMRealm` property to the same value as the other component.
5. Stop the server by using the following command:

   UNIX **`stopWas.sh`**

   Windows
   `stopWas.bat`

   To stop the server, use the WebSphere Application Server administration credentials. For more information about this command, see the section about application server, starting and stopping in *IBM Workload Scheduler: Administration Guide*.
6. Load the new properties by entering the following command:

   UNIX **`changeSecurityProperties.sh`** < *text_file*

   Windows
   `changeSecurityProperties.bat` < *text_file*
7. Restart the server by entering the following command:

   UNIX **`startWas.sh`**

   Windows
   `startWas.bat`

   For more information about this command, see the section about application server, starting and stopping in *IBM Workload Scheduler: Administration Guide*.

Configuring the use of Lightweight Third-Party Authentication

About this task

The WebSphere Application Server uses the Lightweight Third-Party Authentication (LTPA) mechanism to propagate user credentials.

Depending on your circumstances, you might need to configure the use of the same LTPA token_keys between Dynamic Workload Console and the engine, or disable the automatic generation of the LTPA token_keys, or both:

Configuring for Single Sign-On
If you are configuring for Single Sign-On, between any version of Dynamic Workload Console and any engine, whether or not they are installed on the same system, you must configure both instances of WebSphere Application Server involved to use the same LTPA token_keys, and disable their automatic regeneration on expiry, following the procedures described in:

- "Configuring to use the same LTPA token_keys" on page 119
- "Disabling the automatic generation of LTPA token_keys" on page 121

No Single Sign-On, and only one instance of WebSphere Application Server on a system
No action need to be taken.

Configuring to use the same LTPA token_keys

About this task

To use the same LTPA token_keys between more than one WebSphere Application Server, you must run this procedure between Dynamic Workload Console and each engine you want to which you want to connect.

The LTPA token_keys can be either exported from Dynamic Workload Console and imported into the engine, or exported from the engine and imported into Dynamic Workload Console.

Procedure

1. Use the following script to export the LTPA token_keys from the WebSphere Application Server where the Dynamic Workload Console is installed, and to import them into the other instance of WebSphere Application Server. The script is located in the following path:

   IBM Workload Scheduler
   `<TWA_home>/wastools/manage_ltpa.sh` or `...\manage_ltpa.bat`

   Dynamic Workload Console
   `<Dynamic_Workload_Console_install_directory>/wastools/manage_ltpa.sh` or `...\manage_ltpa.bat`, for example, `/opt/IBM/TWAUI/wastools/manage_ltpa.sh`

   There is also a copy of `manage_ltpa.sh` and `manage_ltpa.bat` on each installation image.

   Make sure that the user who runs this script is allowed to access the WebSphere Application Server profile that hosts the Dynamic Workload Console or the engine.

   The syntax used to run the script is the following:

   ```
   manage_ltpa -operation import|export -profilepath profile_path
               -ltpafile LTPA_file_path -ltpapassword LTPA_file_password
               [-user username -password password]
               -port SOAP_port -server server_name
   ```

   where:

   –operation
   Select *export* to read the LTPA token_keys from the profile and save it to a file. Select *import* to update the profile with the LTPA token_keys stored in a file.

   –profilepath
   Specify the path to the profile on top of which the application, either Dynamic Workload Console or IBM Workload Scheduler is installed.

   –ltpafile
   Specify the fully qualified path name of the file that contains, if you import, or where to encrypt, if you export, the LTPA token_keys.

   –ltpapassword
   Specify a password of your choice to encrypt the file that contains the LTPA keys when exporting them, or, when importing them, the password that was used to encrypt them when they were exported. This password is used only when importing and exporting that LTPA token_keys. It does not need to match the administrator password.

   –user The administrator of the server hosting the Dynamic Workload Console

or the engine. In the case of IBM Workload Scheduler, the administrator is, by default, the owner of the instance (*TWS_user*). The user and password arguments are optional. By default, the script looks for the credentials in the `soap.client.props` file located in the properties directory of the WebSphere Application Server profile.

–password
The password of the administrator of the server defined in the selected profile. The user and password arguments are optional. By default, the script looks for the credentials in the `soap.client.props` file located in the properties directory of the WebSphere Application Server profile.

–port Specify the SOAP port used by the profile. By default the SOAP port is 28880 for Dynamic Workload Console installed on the WebSphere Application Server, and 31118 for IBM Workload Scheduler installed on the WebSphere Application Server.

–server
Specify the name of the server of the profile on which to import or export the LTPA tokens. The default server name varies, depending on how it was installed. See Table 28.

Note:

a. The server and path might have been modified from the default value after installation.
b. This keyword is mandatory if the IBM Workload Scheduler server name is different from the Dynamic Workload Console server name.

Table 28. Product versions and default server names

Product version	WebSphere Application Server version	Default server name
IBM Workload Scheduler, V9.*x*:	The WebSphere Application Server installed in an instance of IBM Workload Automation (on which any IBM Workload Scheduler component is installed).	**server1**, found in the following path: `<WAS_profile_path>/config/cells/TWSNodeCell/nodes/TWSNode/servers/server1/server.xml` where the default value of *WAS_profile_path* is `<TWA_home>/WAS/TWSprofile`

	Your version of the WebSphere Application Server on which the Dynamic Workload Console is installed.	**server1**, found in the following path:
		`JazzSM_profile_dir/config/cells/JazzSMNode01Cell/nodes/JazzSMNode01/servers/server1/`
		
		where, the default value of *JazzSM_profile_dir* is:
		On Windows operating systems
		`C:\Program Files\IBM\JazzSM\profile`
		On UNIX operating systems
		`/opt/IBM/JazzSM/profile`

2. Stop and start each server involved in this activity to enable it.
3. If you are configuring Single Sign-On, test that the configuration is correctly set between and the engine by performing the following steps:
   a. Log in to Dynamic Workload Console.
   b. Create an engine connection without specifying User ID and password.
   c. Perform a test connection.

Results

The next step is to disable the automatic generation of the LTPA token_keys, for which see: "Disabling the automatic generation of LTPA token_keys"

Disabling the automatic generation of LTPA token_keys

About this task

Disable the automatic generation of LTPA token_keys if you are enabling Single Sign-On. You must disable the generation of the keys at both ends of the communication, in other words, at the Dynamic Workload Console, and at the engine of IBM Workload Scheduler or dynamic workload broker, as appropriate:

At the Dynamic Workload Console

1. Log in to Dynamic Workload Console.
2. Click **Settings > WebSphere Administrative Console > Launch WebSphere > Administrative Console**.
3. On WebSphere Administrative Console, click **SSL certificate and key management**.
4. Click the **Key set groups** link.
5. Click the name of the key set group displayed in the list.
6. Clear the **Automatically generate keys** check box.
7. Click **OK**.
8. Check in the list that the field **Automatically generate keys** beside the available key set group is set to *false*.

At IBM Workload Scheduler

The implementation of the WebSphere Application Server on the IBM Workload Scheduler engine includes a limited functionality version of the Integrated Solutions Console. Use this portal to disable the automatic generation of LTPA token_keys, as follows:

1. Connect to the Integrated Solutions Console from an Internet browser, using the following URL:

   `http://TWS_server_hostname:WAS_admin_host_(secure_)port/ibm/console`

   Use the **showHostProperties** tool to identify the *WAS_admin_host_port* (default 31123) or *WAS_admin_host_secure_port* (default 31124), as appropriate. For more information about this tool, refer to "Application server - using the utilities that change the properties" on page 459.
2. Perform the procedure you used above to disable the token-keys generation for the Dynamic Workload Console, starting from step 2.

Configuring Dynamic Workload Console to use SSL

The Secure Sockets Layer (SSL) protocol provides secure communications between remote server processes or applications. SSL security can be used to establish communications inbound to, and outbound from, an application. To establish secure communications, a certificate, and an SSL configuration must be specified for the application.

Full details are supplied in "Scenario: Connection between the Dynamic Workload Console and the IBM Workload Scheduler component that has a distributed connector" on page 304.

Creating a Windows service for Jazz for Service Management

Optionally create a service for Jazz for Service Management on Windows systems

About this task

The Dynamic Workload Console requires Jazz for Service Management as a prerequisite. On Windows systems, you can optionally create a service for Jazz for Service Management, using the **smsetApplicationServer** command, as follows:

Procedure

1. Open a command prompt and browse to the *<JazzSM_install_directory>*\bin directory.
2. Type the following command:

   ```
   smsetApplicationServer.bat --property autostart=true --property admin.name=
   <JazzSM_username>
                              --property admin.password=
   <JazzSM_password>
   ```

   where:

   autostart
   When the property is set to `true`, the command can complete some or all of the following actions:
   - Creates the service if it does not exist.
   - Starts the service if it is not started.

   When the property is set to `false`, the command can complete some or all of the following actions:
   - Checks the status of the service.
   - Stops the service if it is not stopped.
   - Deletes the service if it exists.

   admin.name
   Is the user name with administrative privileges who can change the configuration settings of the Jazz for Service Management application server

   admin.password
   Is the password that is associated with the administrator user name as set by the **admin.name** property

Customizing your global settings

How to customize global settings.

About this task

To customize the behavior of the Dynamic Workload Console, you can optionally configure some advanced settings. These settings are specified in a customizable file named `TdwcGlobalSettings.xml.template`.

By default, the customizable file is copied into the following path after you install the Dynamic Workload Console:

For Windows systems:

```
C:\Program Files\IBM\JazzSM\profile\registry\
TdwcGlobalSettings.xml.template
```

For UNIX and Linux systems:

```
/opt/IBM/JazzSM/profile/registry/TdwcGlobalSettings.xml.template
```

You can find a copy of this file also on the installation media in the directory `/utilities/TdwcGlobalSettings.xml`.

If you have Administrator privileges, you can modify the file to replace default values with customized ones and enable commented sections. To enable commented sections, remove the `<!--` and `-->` tags that enclose the section. You then save the file locally with the name `TdwcGlobalSettings.xml`.

You can add and modify some customizable information, such as:

| • The URLs that link to videos in the Dynamic Workload Console. For example, you can link to a company intranet server to view help videos rather than to a public video site.
- The maximum number of objects to be shown in the graphical views.
- The setting to display the plan view in a new window.
* • The auto refresh interval for the **Show Plan View** graphical view.
- The configuration details to enable the news notification beacon and be constantly up-to-date with product information. See Disabling news notification.
- The creation of predefined tasks.
- The URLs where you can store customized documentation about your jobs or job streams to associate customized documentation to them.
- The current user registry in use.
- The timeout to read and write information on a IBM Workload Scheduler for z/OS engine.
- The maximum number of objects to be retrieved with a query, the maximum number of rows to display in a table, and the maximum number of direct queries to maintain in history.
- Allowing or preventing users from sharing tasks and engine connections.
- The display of all dependencies, both satisfied and unsatisfied.
- The use of audit files to track activities in the Self-Service Catalog and Self-Service Dashboards mobile applications.
= • Displaying or hiding all predecessors from the What-if Analysis Gantt view.

This file is accessed at each login, and all configurations specified in the file are immediately applied, except for the **precannedTaskCreation** property. This property is read only when a user logs in for the first time and is then used whenever this user logs in again.

You can use any text or XML editor to edit this file, but ensure that you save it is as a valid XML file.

The file is organized into sections that group similar properties. An explanation of each section is available in the file. For more information, see "TdwcGlobalSettings.xml sample" on page 134.

Sections can also be repeated multiple times in the same file and applied differently to different user roles. To apply a section only to the users belonging to a role, the section must be included within the tags `<settings role="user_role">` and `</settings>`, where:

<user_role>
The user for which the enclosed configuration must be applied. The default value is all users, unless otherwise specified.

Only one **settings** section can be specified for each role. If a user has more than one role, the settings associated to the higher role are used.

Example:

```
<?xml version"1.0"?>
<tdwc>
.
.
<settings>
<graphViews>
<property name="planViewNewWindow" value="true"/>
</graphViews>
</settings>

<settings  role="TWSWEBUIOperator">
<graphViews>
<property name="planViewNewWindow" value="false"/>
</graphViews>
</settings>
.
.
</tdwc>
```

To view the complete syntax for the file, see "TdwcGlobalSettings.xml sample" on page 134.

Customize video URLs

This section shows how you should customize your URLs that link video content in the Dynamic Workload Console so that you can link to a company intranet server to view help videos rather than a public video site.

The `_baseURL` prefix will be added to all your video URLs . If you do not specify a link for your video the default setting will automatically be used.

```
<codeblock><?xml version"1.0"?>
<tdwc>
.
.
<settings>
-<videoGallery>
<property name="_baseURL" value=""></property>
<property name="depLoop" value=""></property>
<property name="highlightRelDep" value=""></property>
<property name="viewDepPrompt" value=""></property>
<property name="usingImpactView" value=""></property>
<property name="createUseTasks" value=""></property>
<property name="weAddRemoveFile" value=""></property>
<property name="weCreateDeps" value=""></property>
<property name="weAddJob" value=""></property>
<property name="weHighlightDeps" value=""></property>
<property name="weCreateJCL" value=""></property>
</videoGallery>
```

```
<!-- </settings>
.
.
</tdwc>
```

Override graphical view limits

This section contains the configuration parameters that apply to the graphical views in the plan, such as the maximum number of objects shown in each view.

planViewMaxJobstreams
The maximum number of job streams displayed in the Plan View. Default value is **1000**. Values greater than **1000** are not supported.

preProdPlanViewMaxJobstreams
The maximum number of job streams displayed in the preproduction plan view. Default value is **1000**. Values greater than **1000** are not supported.

```
<?xml version"1.0"?>
<tdwc>
.
.
 <settings>
<graphViews>
<property name="planViewMaxJobstreams" value="1000"></property>
<property name="preProdPlanViewMaxJobstreams" value="1000"></property>
</graphViews>
 </settings>.
.
</tdwc>
```

See "TdwcGlobalSettings.xml sample" on page 134 to view the complete syntax for the file.

Plan View in new window

This section is used to prevent Internet Explorer 7 from freezing while using the Plan View. To solve the problem, set value to **true**.

planViewNewWindow
Set it to **true** if you want the plan view to be displayed in a new window each time it is launched. Default value is **false**.

```
<?xml version"1.0"?>
<tdwc>
.
.
 <settings>
<graphViews>
<property name="planViewNewWindow" value="true"/>
</graphViews>
.
.
 </settings>
</tdwc>
```

See "TdwcGlobalSettings.xml sample" on page 134 to view the complete syntax for the file.

Plan View auto refresh interval

Use this section to change the default setting of the auto refresh interval for the Show Plan View graphical view for all users. By default, the auto refresh interval is `300 seconds` (five minutes).

PlanViewAutorefresh

The graphical representation of the Plan View is automatically refresh every 300 seconds by default. To change this setting, edit the value assigned to the **DefaultTime** property. The minimum value you can set is 30 seconds. Any value specified below this value is reset to 30 seconds. You must restart the Dynamic Workload Console application server after modifying this value.

```
<?xml version"1.0"?>
<tdwc>
.
.
 <settings>
<PlanViewAutorefresh>
<property name="DefaultTime" value="300"/>
</PlanViewAutorefresh>
.
.
 </settings>
</tdwc>
```

See "TdwcGlobalSettings.xml sample" on page 134 to view the complete syntax for the file.

Disable and customize NewsFeed function

This section contains the configuration details to be constantly up-to-date with product information.

FeedURL

Contains the URL from which you receive news and updates. Default value is: https://www.ibm.com/developerworks/community/wikis/form/anonymous/api/wiki/585f5525-a7f5-48ef-9222-50ad582e85f4/page/e599dd3c-8dc3-4ab6-89fd-33f81a994799/attachment/de677e63-5a9d-46db-a010-18ca38f05812/media/tws.jsonp

FeedType

A string that identifies the format of update information. Default value is **JSONP**.

PollInterval

The interval in seconds between two checks for updates. Default value is **600**.

PollInitialDelay

An initial delay in seconds before the first attempt to read the news feeds. After the initial load, the poll interval is used. Default value is **120**.

NewsFeed

Property used to add further customized news feeds. Specify the format and address of the file that contains the customized communication. Supported formats are RSS 2.0 and ATOM 1.0. You must write the communication in ATOM 1.0 or RSS 2.0 format and store this file in the an HTTP server complying with the *same origin policy*. For browser security reasons, this policy permits to access information only on server using the same protocol, hostname and port number as the one to which you are connected. Optionally, if you want to store your customized feed on an external server, you must configure an HTTP reverse proxy server mapping the external server address.

```
<property name="NewsFeed" type="RSS"
value="http://DWC_hostname:portnumber.com/news.rss" />
```

Note: To specify multiple feeds, you must specify multiple **NewsFeed** properties.

NewsFeedCategory
The name of the customized information. It can be used to identify informational, warning or alert messages, for example. The path to an image can also be added to better identify the information with an icon.

To add more category images, specify a list of properties named **NewsFeedCategory**, for example:

```
<property name="NewsFeedCategory" value="my company info"
icon="http://www.my.company.com/info.png" />
<property name="NewsFeedCategory" value="my company alert"
icon="http://www.my.company.com/alert.png" />
```

If no customized feed is specified, the default feed is used, which retrieves the latest product information from official support sites. To disable any notification, comment the entire section. To disable only external notifications about product information updates, assign an empty string as value to the `FeedURL` property of `JSONP` feed like:

```
  <property name="FeedURL" type="JSONP" value="" />
```

Example:

```
<?xml version"1.0"?>
<tdwc>
.
.
 <settings>
<NewsFeed>
<property name="NewsFeed" type="RSS"
value="http://www.DWC_hostname:portnumber.com/my_rss.xml" />
<property name="NewsFeed" type="ATOM"
value="http://www.DWC_hostname:portnumber.com/my_atom.xml" />

<property name="PollInterval" value="600" />
<property name="PollInitialDelay" value="1" />

<property name="FeedURL" type="JSONP" value="" />

<property name="NewsFeedCategory"
value="my company info" icon="http://www.DWC_hostname:portnumber.com
/info.png" />
<property name="NewsFeedCategory"
value="my company alert" icon="http://www.DWC_hostname:portnumber.com
/alert.png" />

</NewsFeed>
 </settings>
.
.
</tdwc>
```

See “TdwcGlobalSettings.xml sample” on page 134 to view the complete syntax for the file.

Disable and customize the creation of predefined tasks

This section defines the environment for which predefined tasks are created.

precannedTaskCreation
Some predefined tasks are created by default and are available when you log in to the console. There is a predefined Monitor task for every object,

for both z/OS and distributed engines. Default value is **all**. To change this setting, use one of the following values:

all All predefined tasks are created. This is the default.

distributed
Only predefined tasks for distributed engines are created

zos Only predefined tasks for z/OS engines are created

none No predefined task is created.

```
<?xml version"1.0"?>
<tdwc>
.
.
 <settings>
<application>
<property name="precannedTaskCreation" value="all"/>
</application>
 </settings>
.
.
</tdwc>
```

See "TdwcGlobalSettings.xml sample" on page 134 to view the complete syntax for the file.

Add customized URL to job and job streams

This section contains URLs where you can store customized documentation about your jobs or job streams. By default, this setting is not specified. If you want to associate customized documentation to a job or job stream, use this setting to specify the external address where this information is located.

If you want to specify a URL where customized documentation for a job and job stream is stored, uncomment the section lines, specify the required URL, and optionally assign a name to the UI label by specifying a value for the customActionLabel property. By default this name is **Open Documentation**. This label is then displayed in the **More Actions** menus in Monitor Jobs and Monitor Job Streams tasks, as well as in the graphical views of the plan (in the object's tooltips, context menus and properties). In this example, selecting **Open Documentation** accesses the relevant documentation making it possible to open the documentation while monitoring your job or job stream in the plan.

To implement this setting, assign values to the following keywords:

customActionLabel
The name of the action displayed in menus, object properties, and tooltips to access customized documentation about your jobs or job streams. By default this name is "Open Documentation" unless you customize the name with this keyword.

jobUrlTemplate
The address of your job documentation. No default value available.

jobstreamUrlTemplate
The address of your job stream documentation. No default value available.

```
<?xml version"1.0"?>
<tdwc>
.
.
 <settings>
```

```
  <twsObjectDoc>
    <property name="jobstreamUrlTemplate"
      value="http://www.yourhost.com/tws/docs/jobstream/${js_name_w}"/>
    <property name="jobUrlTemplate"
      value="http://www.yourhost.com/docs/jobs/${job_name_w}"/>
    <property name="customActionLabel" value="Your Custom Label Name"/>
  </twsObjectDoc>
 </settings>
.
.
</tdwc>
```

See "TdwcGlobalSettings.xml sample" on page 134 to view the complete syntax for the file.

These properties must be valid URLs, containing one or more of the variables listed in the table below.

If you use any of the following special characters in the URL, you must write them as follows:

Table 29. Syntax for special characters

Special characters	**Write them as...**
quote (")	\"
apostrophe (')	&apos;
ampersand (&)	&amp;
less than (<)	&lt
greater than (>)	&gt
backslash (\)	\\

Multiple variables can be included in a URL and must be specified using the following syntax: ${*variable*}:

Table 30. Variables used in the URL definition

Name	**Object**	**Description**
job_number_w	Job z/OS	The number of the job
job_wkst_w	Job	The name of the workstation on which the job runs
job_jsname_w	Job	The name of the job stream that contains the job
job_jswkst_w	Job	The name of the workstation on which the job stream runs
job_actualarrival_w	Job z/OS	The actual start time of the job (date format: YYYY-MM-DDThh:mm:ss)
job_actualend_w	Job z/OS	When the job actually completed (date format: YYYY-MM-DDThh:mm:ss)
job_starttime_w	Job	The start time of the job (date format: YYYY-MM-DDThh:mm:ss)
job_id_w	Job	The ID of the job

Table 30. Variables used in the URL definition (continued)

Name	Object	Description
job_returncode_w	Job	The return code of the job
js_name_w	Job stream	The name of the job stream that contains the job
js_wkst_w	Job stream	The name of the workstation on which the job stream runs
js_id_w	Job stream	The job stream ID
js_latest_start_w	Job stream	The latest time at which a job stream can start (date format: YYYY-MM-DDThh:mm:ss)
engine_name_w	Engine	The name of the engine connection
engine_host_w	Engine	The hostname of the engine connection
engine_port_w	Engine	The port number of the engine connection
engine_plan_w	Engine	The ID of selected plan
engine_serv_w	Engine	The remote server name of the engine connection

User registry

Use this section to configure some properties related to the User Registry in use.

groupIdMap

The property is related to the groups of User Registry, and can be modified to map and display the specified value of each group. The default is the common name of the group.

Examples:

```
<?xml version"1.0"?>
<tdwc>
.
.
<settings>
<security>
<property name="groupIdMap" value="cn"></property>
</security>
</settings>
.
.
</tdwc>
```

Therefore, if you need to change the default value "cn" to "racfid", you can define this property as follows:

```
<property name="groupIdMap" value="racfid"></property>
```

See "TdwcGlobalSettings.xml sample" on page 134 to view the complete syntax for the file.

z/OS http connections

Use this section to configure the timeout to read and write information on IBM Workload Scheduler for z/OS engine. When you connect to the IBM Workload Scheduler for z/OS engine to retrieve a list of defined objects, you receive an error message if the list is not returned within the timeout period. The value is expressed in milliseconds.

Example:

```
<?xml version"1.0"?>
<tdwc>
.
.
 <settings>
<http>
<property name="zosHttpTimeout" value="90000" />
</http>
.
.
 </settings>
</tdwc>
```

See "TdwcGlobalSettings.xml sample" on page 134 to view the complete syntax for the file.

Limit the number of objects retrieved by queries

If you are connected to engines V9.1 this setting is ignored.

Use this section to configure: the number of results displayed for Monitor tasks, the maximum number of rows to display on each page, and the number of direct queries to maintain in history.

If you want to limit the number of results produced by your queries, you can specify the maximum number of items that must be retrieved using the monitorMaxObjectsPM property. The minimum number of retrieved results is 500.

The default value is -1; any value lower than 0 means that there is no limit in the number of objects retrieved.

Because data is extracted in blocks of 250 rows, the value you enter is adjusted to complete an entire block. For example, if you specify a limit of 500, only 500 elements are retrieved, while if you specify a limit of 600, 750 elements are retrieved.

For Multiple engine tasks, this limit is applied to each engine included in the query. Therefore, if you specify a limit of 500 results and, for example, you run a Monitor jobs on multiple engine task on three engines, the results produced by your query will be no more than 500 *for each engine*, for a maximum of 1500 rows.

Note: This setting does not apply to Monitor critical jobs tasks.

To set the maximum number of rows to display in a table view, configure the maxRowsToDisplay property.

To set the maximum number of direct queries to maintain in history, configure the maxHistoryCount property. These queries are available from the pull-down for the Query field on the Monitor Workload page.

```
<?xml version"1.0"?>
<tdwc>
.
.
 <settings>
  <monitor>
    <property name="monitorMaxObjectsPM" value="2000"></property>
  </monitor>

  <monitor>
    <property name="maxRowsToDisplay" value="25"></property>
  </monitor>

  <monitor>
    <property name="maxHistoryCount" value="100"></property>
  </monitor>
 </settings>
.
.
</tdwc>
```

See "TdwcGlobalSettings.xml sample" on page 134 to view the complete syntax for the file.

Limit task and engine sharing

Use this section to prevent users from sharing tasks and engines.

By default there is no limit to task and engine sharing and all users are authorized to share their tasks and engine connections. If you want to change this behavior, preventing users from sharing tasks and engines, set this property to **true**.

The property default value is **false**, set it to **true** to enable the limit:

limitShareTask
Set to true to prevent users from sharing tasks.

limitShareEngine
Set to true to prevent users from sharing engine connections.

```
<?xml version"1.0"?>
<tdwc>
.
.
 <settings>
  <security>
   <property name="limitShareTask"     value="false" />
   <property name="limitShareEngine"   value="false" />
  </security>
 </settings>
.
.
</tdwc>
```

See "TdwcGlobalSettings.xml sample" on page 134 to view the complete syntax for the file.

Show all dependencies

This section defines whether to show all dependencies displayed, regardless of their being satisfied or not.

ShowDependencies
When you open the dependencies panel from Monitor jobs and Monitor job streams task results, by default only **Not Satisfied** dependencies are

shown. Uncomment this section and leave the value set to "**true**" to have all dependencies displayed, regardless of their being satisfied or not. Possible values are:

true All dependencies displayed, regardless of their being satisfied or not.

false Only not satisfied dependencies are displayed.

```
<?xml version"1.0"?>
<tdwc>

.
.
 <settings>
  <ShowDependencies>
   <property name = "AlwaysShowAllDependencies"
             value="true"></property>
  </ShowDependencies>
 </settings>
.
.
 </tdwc>
```

See "TdwcGlobalSettings.xml sample" on page 134 to view the complete syntax for the file.

Auditing mobile app activity

This section defines whether to track activities performed in the Self-Service Catalog and Self-Service Dashboards applications in an auditing log file.

For information about the name and location of the log file, see the logs and traces section in the *Troubleshooting Guide*.

SSAuditing
This value is set to "**true**" by default so that operations performed in the Self-Service Catalog and Self-Service Dashboards applications are written to a log file. The log file contains information such as creation, modification and deletion dates, the operations performed in the mobile apps, and the user performing the operations. Possible values are:

true Operations performed in the Self-Service Catalog and Self-Service Dashboards applications are tracked in an auditing log file.

false Operations performed in the Self-Service Catalog and Self-Service Dashboards applications are not tracked in an auditing log file.

SSAuditingLogSize
The maximum size of a log file in KB. When a log file reaches the maximum size, the system rolls that log file over and creates a new file. By default, the maximum size of a log file is 100 KB.

SSAuditingLogFiles
The default number of log files to create. When this number is met and the latest log file reaches its maximum size, the system deletes the oldest log file and rolls the latest file over and creates a new file.

```
<?xml version"1.0"?>
<tdwc>
.
.
 <settings>
 <SSCMAuditing>
            <property name = "SSAuditing"         value="true"></property>
```

```
            <property name = "SSAuditingLogSize"  value="100"></property>
            <property name = "SSAuditingLogFiles" value="2"</property>
 </settings>
 .
 .
</tdwc>
```

See “TdwcGlobalSettings.xml sample” to view the complete syntax for the file.

Modifying the number of archived plans displayed in the Dynamic Workload Console

You can modify the number of archived plans displayed in the Monitor Workload view of the Dynamic Workload Console. The default number is 30 plans.

To modify the default number, configure the following property in the **TdwcGlobalSettings.xml** file:

```
<monitor>
    <property name="maxArchivedPlan"value="30"></property>
  </monitor>
```

See “TdwcGlobalSettings.xml sample” to view the complete syntax for the file.

Show or hide predecessors from What-if Analysis Gantt view

When you have hundreds of predecessors, you can optimize performance by excluding them from the What-if Analysis Gantt view. By default, all predecessors are loaded into the What-if Analysis Gantt view. To exclude them, uncomment this section and leave the default setting of the property **whatIfAutoLoadPreds** to "false" . To revert back to the default behavior either set the property to "true" or comment the section again in the **TdwcGlobalSettings.xml** file.

To modify the default setting, configure the following property in the **TdwcGlobalSettings.xml** file:

```
      <WhatifAnalysis>
            <property name = "whatIfAutoLoadPreds" value="false"></property>
      </WhatifAnalysis>
```

See “TdwcGlobalSettings.xml sample” to view the complete syntax for the file.

TdwcGlobalSettings.xml sample

The following example is a sample of the file:

```
 <?xml version="1.0"?>
 <tdwc>
################################################################################
################  SETTINGS FOR ALL USERS                  ######################
################################################################################
<settings>
 ################################################################################
 ## CUSTOMIZE LINKS TO VIDEOS ###################################################
 ################################################################################
 -->
<!--
 This section shows how you should customize your URLs that link video content in
 the Dynamic Workload Console so that you can link to a company intranet server to
 view help videos rather than a public video site.
 -->
<!--
 <videoGallery>
 <property name="_baseURL" value=""></property>
 <property name="depLoop" value=""></property>
 <property name="highlightRelDep" value=""></property>
 <property name="viewDepPrompt" value=""></property>
 <property name="createUseTasks" value=""></property>
 <property name="weAddRemoveFile" value=""></property>
 <property name="weCreateDeps" value=""></property>
 <property name="weAddJob" value=""></property>
 <property name="weHighlightDeps" value=""></property>
 <property name="weCreateJCL" value=""></property>
```

```
 </videoGallery>
-->
<!--
################################################################################
#################  SECTION 1 - GRAPHICAL VIEW SETTINGS  ########################
################################################################################
-->
<!--
 This section specifies the maximum number of objects shown in each graphical view.
 Default value is 1000 for all properties. Values greater than 1000 are not supported.
  -->
<!--
  <graphViews>
     <property name="planViewMaxJobstreams" value="1000"></property>

     <property name="preProdPlanViewMaxJobstreams" value="10"></property>


  </graphViews>

################################################################################
###################  SECTION 2 - PLAN VIEW IN NEW WINDOW  ######################
################################################################################
-->
<!--
This section is used to prevent Internet Explorer 7 from freezing while using the Plan View.
 To solve the problem, set value to true. Default value is false.
-->
<graphViews>
<property name="planViewNewWindow" value="true"/>
</graphViews> <!--

      ################################################################################
###################  SECTION 3 - PLAN VIEW AUTO REFRESH   ######################
################################################################################
-->
<!--
Use this section to change the default setting of the auto refresh interval for the
     Show Plan View graphical view for all users. By default, the auto refresh interval is
     300 seconds (five minutes). The minimum value you can set is 30 seconds.
     Any value specified below this value is reset to 30 seconds. You must restart the
     Dynamic Workload Console application server
     after modifying this value.
-->
<PlanViewAutorefresh>
      <property name="DefaultTime" value="300"></property>
      </PlanViewAutorefresh> <!--

################################################################################
################  SECTION 4 - DISABLE / CUSTOMIZE NEWS FEED FUNCTION  ##########
################################################################################
-->
<!--
This section allows overriding the properties concerning the "NewsFeed" function.
Default values are as follows:
<NewsFeed>
<property name="FeedURL" value="https://www.ibm.com/developerworks/community/wikis/form/
   anonymous/api/wiki/585f5525-a7f5-48ef-9222-50ad582e85f4/page/e599dd3c-8dc3-4ab6-89fd-
   33f81a994799/attachment/de677e63-5a9d-46db-a010-18ca38f05812/media/tws.jsonp"
<property name="FeedType" value="JSONP" />
<property name="PollInterval" value="3600" />
</NewsFeed>
-->
<!--
To disable the function
-->
<!--
<NewsFeed>
<property name="FeedURL" value="" />
<property name="FeedType" value="JSONP" />
<property name="PollInterval" value="3600" />
</NewsFeed>
-->
<!--
################################################################################
############  SECTION 5 - DISABLE /CUSTOMIZE CREATION OF PREDEFINED TASKS  ######
################################################################################
-->
<!--
To avoid or customize the creation of predefined tasks at first logon.
Possible values are:
all             both distributed and z/OS tasks are created. This is the default value
none            no task is created
distributed     only distributed tasks are created
zos             only z/OS tasks are created
-->
<!--
<application>
<property name="precannedTaskCreation" value="all"/>
  </application>
-->

<!--
################################################################################
############  SECTION 6 - ADD A CUSTOM DOCUMENTATION URL TO JOB/JOB STREAM  ######
################################################################################
-->
<!--
This section contains URLs where you can store customized documentation about your jobs
 or job streams. By default this setting is not specified. If you want to associate
 customized documentation to a job or job stream, use this setting to specify the
 external address where this information is located. If you want to specify a URL to be
 opened as related documentation for job and job stream, uncomment the section lines so
 that a new action, Open Documentation, is inserted in the More Actions menu for Monitor
 Jobs and Monitor Job Streams tasks. The new action links to the specified URL
```

```
You can customize the URL template by using variables. The variables have the syntax
                    ${<variable_name>}

For the complete list of variables, see the documentation.

-->
<!--
  <twsObjectDoc>
<property name="jobstreamUrlTemplate" value="http://www.yourhost.com/tws/docs/jobstream/${js_name_w}" />
<property name="jobUrlTemplate"    value="http://www.yourhost.com/docs/jobs/${job_name_w}" />
  <property name="customActionLabel"    value="Custom Action" />
</twsObjectDoc>
-->

<!--
###############################################################################
############  SECTION 7 - USER REGISTRY                          ##########
###############################################################################
  In this section you can configure some properties about the User Registry in use.
  The property groupIdMap is related to the groups of User Registry, and can be modified
  to map and display the specified value of each group. By default the common name of the
       group is displayed.
-->
<!--
<security>
<property name="groupIdMap" value="cn"></property>
</security>
    -->
<!--
   ###############################################################################
############  SECTION 8 - Z/OS HTTP CONNECTIONS                          ##########
###############################################################################

Use this section for increase or decrease timeout for http connection in Z/OS
environment. Change this setting if you receive a connection timeout using plugin
      actions/picklists.

The setting is in milliseconds.
-->
 <!--
<http>
<property name="zosHttpTimeout" value="90000" />
</http>
    -->
<!--
  ###############################################################################
########  SECTION 9 -  LIMIT THE NUMBER OF OBJECTS RETURNED BY THE QUERIES ##########
###############################################################################
<!--
Use this section to configure: the number of results displayed for Monitor tasks, the
 maximum number of rows to display on each page, and the number of direct queries to
 maintain in history. This setting applies to all tasks except for Monitor critical jobs
 and Monitor jobs on multiple engines.
 If you want to limit the number of results produced by your queries, you can specify the
 maximum number of items that must be retrieved. The default value is -1; any value lower
 than 0 means that there is no limit in the number of objects retrieved. The minimum number
 of retrieved results is 500. Because data is extracted in blocks of 250 rows, the value
 you enter is adjusted to complete an entire block. For example, if you specify a limit of
 500, only 500 elements are retrieved, while if you specify a limit of 600, 750 elements
 are retrieved.
 To set the maximum number of rows to display in a table view, configure the
 maxRowsToDisplay property.
 To set the maximum number of direct queries to maintain in history, configure the
 maxHistoryCount property. These queries are available from the pull-down for the Query
 field on the Direct Query page.

<monitor>
   <property name="monitorMaxObjectsPM" value="2000"></property>
 </monitor>

 <monitor>
   <property name="maxRowsToDisplay" value="25"></property>
 </monitor>

 <monitor>
   <property name="maxHistoryCount" value="100"></property>
 </monitor>
-->


<!--

  ###############################################################################
########  SECTION 10 -  LIMIT TASK AND ENGINE SHARING                      ##########
###############################################################################

Use this section to prevent users from sharing tasks and engines.
 By default there is no limit to task and engine sharing and all users are authorized to share
 their tasks and engine connections. If you want to change this behavior, preventing users from
 sharing tasks and engines, set this property to true. The property default value is false,
 set it to true to enable the limit:

-->
<!--
<security>
<property name="limitShareTask"     value="false" />
<property name="limitShareEngine"   value="false" />
  </security>
-->


<!--

###############################################################################
########  SECTION 11 -  CHANGE DEFAULT BEHAVIOR FOR DEPENDENCIES PANEL     ##########
###############################################################################
```

```
 Use this section to change the default behavior of the UI when displaying dependencies in
 the dependencies panel. By setting this value to true, by default, all dependencies are
 displayed, and not just the unsatisfied ones.
-->
<!--
 <ShowDependencies>
       <property name = "AlwaysShowAllDependencies"
 value="true"></property>
    </ShowDependencies>
-->

    ##############################################################################
    ########  SECTION 12 -  CHANGE DEFAULT BEHAVIOR FOR SSC AND SSD AUDITING   ###########
    ##############################################################################

   Use this section to change the default behavior of the auditing of activities performed
    using the Self-Service Catalog and the Self-Service Dashboards applications. By default,
    auditing is enabled. You can also set the maximum size of the log file before it rolls
    over to a new log file, and the maximum number of log files maintained.
-->
   <!--  <SSCMAuditing>
           <property name = "SSAuditing"         value="true"></property>
           <property name = "SSAuditingLogSize"  value="100"></property>
           <property name = "SSAuditingLogFiles" value="2"</property>
-->


    ##############################################################################
    ########  SECTION 13 -  URL FOR AGENT LICENSE                            ###########
    ##############################################################################

   Use this section to change the default Agent License URL.
-->
   <!--     <AgentLicense>
           <property name = "URL" value="https://controller.wa.ibmserviceengage.com/SaaSGovernorWeb
           /LicenseServlet"></property>
           </AgentLicense>##############################################################################################
    ########  SECTION 14 -               WHAT-IF ANALYSIS                    ###########
    ##############################################################################

  Use this section to show or hide predecessors from the What-If Analysis Gantt view.
        By default, all predecessors are loaded in the view. To exclude them, uncomment this
        section and leave the setting of the property whatIfAutoLoadPreds to
        "false". To revert back to the default behavior, either set the property to "true"
        or comment this section again so that it is ignored.
-->
   <!--     <WhatifAnalysis>
           <property name = "whatIfAutoLoadPreds" value="false"></property>
           </WhatifAnalysis>

-->
           </settings>

 <!--
 ##############################################################################
 ################  SETTINGS FOR ALL TWSWEBUIAdministrators users ###################
 ##############################################################################
 -->
 <settings role="TWSWEBUIAdministrator">
 <!-- Put here setting to be applied only to users with TWSWEBUIAdministrator role  -->
 </settings>
 <!--
 ##############################################################################
 ################  SETTINGS FOR ALL TWSWEBUIOperators users    ###################
 ##############################################################################
 -->
 <settings role="TWSWEBUIOperator">
 </settings>
 <!--
 ##############################################################################
 ################  SETTINGS FOR ALL TWSWEBUIConfigurator users ###################
 ##############################################################################
 -->
 <settings role="TWSWEBUIConfigurator">
 </settings>
 <!--
 ##############################################################################
 ################  SETTINGS FOR ALL TWSWEBUIDeveloper users    ###################
 ##############################################################################
 -->
 <settings role="TWSWEBUIDeveloper">
 </settings>
 <!--
 ##############################################################################
 ################  SETTINGS FOR ALL TWSWEBUIAnalyst users      ###################
 ##############################################################################
 -->
 <settings role="TWSWEBUIAnalyst">
 </settings>

</tdwc>
```

Disable the What-if Analysis

You can disable the What-if Analysis in your environment by setting the **optman enWhatIf | wi** global option to *no* (default value is *yes*).

The `enWhatIf | wi` global option interacts with the `enWorkloadServiceAssurance | wa` global option, which enables or disables privileged processing of mission-critical jobs and their predecessors. For details about this interaction, see the following table.

Table 31. Interaction between `enWorkloadServiceAssurance` and `enWhatIf` global options

Options	Interaction
enWorkloadServiceAssurance | wa is set to *yes* **enWhatIf | wi** is set to *yes*	Both the Workload service assurance and the What-if Analysis features are fully enabled in your environment.
enWorkloadServiceAssurance | wa is set to *yes* **enWhatIf | wi** is set to *no*	The Workload service assurance is enabled. The What-if Analysis feature is disabled and an exception is issued if you try to use it.
enWorkloadServiceAssurance | wa is set to *no* **enWhatIf | wi** is set to *yes*	The Workload service assurance is partially enabled, just to allow the What-if Analysis feature to work properly. This means that: • The Workload service assurance is disabled and an exception is issued if you try to use it. • No critical job is added to the plan.
enWorkloadServiceAssurance | wa is set to *no* **enWhatIf | wi** is set to *no*	Both the Workload service assurance and the What-if Analysis features are disabled in your environment.

Configuring High Availability for Dynamic Workload Console

You can configure a cluster of console nodes in High Availability with identical configurations to evenly distribute user sessions.

Before you begin configuring your nodes in High Availability, refer to the section on configuring High Availability in the *IBM Workload Automation: Dynamic Workload Console User's Guide*.

By leveraging High Availability configuration on Dashboard Application Services Hub, it is possible to meet High Availability requirements for the Dynamic Workload Console. Therefore, the following topics describe how to set up High Availability configuration for Dashboard Application Services Hub, how to customize it for the Dynamic Workload Console, and how to upgrade an existing High Availability configuration on Dashboard Application Services Hub.

High Availability is ideal for Dashboard Application Services Hub installations with a large user population. When a node fails, new user sessions are directed to other active nodes.

You can create a High Availability configuration from an existing stand-alone Jazz for Service Management instance, but must export its custom data before you configure it for High Availability. The custom data is added to the central repository and subsequently replicated to new nodes as they are added to the cluster. The exported data is later imported to one of the nodes in the cluster so that it is replicated across the other nodes in the cluster.

The workload is distributed by session, not by request. If a node fails, users who are in session with that node must log back in to access the Dashboard Application Services Hub. Any unsaved work is not recovered.

Restriction: Before installing a fix pack in a load balanced environment, you must remove all nodes from the load balanced cluster. After removing all nodes from the cluster, you must install the fix pack on each node so that they are at the same release level of Dashboard Application Services Hub. You can recreate the load balanced cluster after updating each node.

Synchronized data

After High Availability is set up, changes in the Dynamic Workload Console that are stored in global repositories are synchronized to all of the nodes in the configuration using a common database. The following actions cause changes to the global repositories used by the Dynamic Workload Console. Most of these changes are caused by actions in the **Settings** folder in the console navigation.

- Creating, restoring, editing, or deleting a page.
- Creating, restoring, editing, or deleting a view.
- Creating, editing, or deleting a preference profile or deploying preference profiles from the command line.
- Copying a portlet entity or deleting a portlet copy.
- Changing access to a portlet entity, page, external URL, or view.
- Creating, editing, or deleting a role.
- Changes to portlet preferences or defaults.
- Changes from the **Users and Groups** applications, including assigning users and groups to roles.

Note: Global repositories must never be updated manually.

During normal operation within a High Availability configuration, updates that require synchronization are first committed to the database. At the same time, the node that submits the update for the global repositories notifies all other nodes in the High Availability configuration about the change. As the nodes are notified, they get the updates from the database and commit the change to the local configuration.

If data fails to be committed on any given node, a warning message is logged in the log file. The node is prevented from making its own updates to the database. Restarting the Dashboard Application Services Hub instance on the node resolves most synchronization issues, if not, remove the node from the High Availability configuration for corrective action.

Note: If the database server restarts, all connections from it to the High Availability configuration are lost. It can take up to five minutes for connections to be restored, and for users to continue performing update operations, for example, modifying or creating views or pages.

Manual synchronization and maintenance mode

Updates to deploy, redeploy, or remove console modules are not automatically synchronized within the High Availability configuration. These changes must be performed manually on each node. For deploy and redeploy operations, the console module package must be identical at each node.

When one of the deployment commands is started on the first node, the system enters *maintenance mode* and changes to the global repositories are locked. After you finish the deployment changes on each of the nodes, the system returns to an unlocked state. There is no restriction to the order that modules are deployed, removed, or redeployed on each of the nodes.

While in maintenance mode, any attempts to make changes in the portal that affect the global repositories are prevented and an error message is returned. Later, the only changes to global repositories that are allowed are changes to a user's personal portlet or widget preferences. Any changes outside the control of the console, for example, a form submission in a portlet to a remote application, are processed normally.

The following operations are also not synchronized within the High Availability configuration and must be performed manually at each node. These updates do not place the High Availability configuration in maintenance mode.

- Deploying, redeploying, and removing wires and transformations
- Customization changes to the Dynamic Workload Console user interface (for example, custom images or style sheets) using `consoleProperties.xml`.

To reduce the chance that users establish sessions with nodes that have different wire and transformation definitions or user interface customizations, schedule these changes to coincide with console module deployments.

Requirements

The following requirements must be met before High Availability can be enabled.

- Install the software requirements:
  1. Install IBM Installation Manager.
  2. Install WebSphere Application Server.
  3. Install Jazz for Service Management with Dashboard Application Services Hub.
  4. Install the Dynamic Workload Console. All the nodes in the High Availability configuration must be at the same release level, must have synchronized clocks, and must be installed in the same cell name. After Dynamic Workload Console installation on each node, use the **-cellName** parameter on the **manageprofiles** command.

     If you are creating a High Availability configuration from a stand-alone instance of the Dynamic Workload Console, you must export its custom data before you configure it for High Availability. The custom data is added to the central repository and subsequently replicated to new nodes as they are added to the High Availability configuration. When you have configured the nodes, you can import the data to one of the nodes for it to be replicated across the other nodes.
  5. Configure the Dynamic Workload Console in LDAP. Lightweight Directory Access Protocol (LDAP) must be installed and configured as the user repository for each node in the High Availability configuration. Each node in the High Availability configuration must be enabled to use the same LDAP using the same user and group configuration. See “Configure the Dynamic Workload Console in LDAP” on page 150.

     For information about which LDAP servers you can use, see List of supported software for WebSphere Application Server V8.5. For information about how to enable LDAP for each node, see Configuring LDAP user registries.

6. Create a new or use an existing database. A supported version of DB2 must be installed within the network to synchronize the global repositories for the nodes defined in the Dynamic Workload Console High Availability configuration. Refer to the System Requirements Document at http://www-01.ibm.com/support/docview.wss?rs=672&uid=swg27048858 for the list of supported database versions. To create a new database, see Creating databases. To use an existing database, see Changing settings repository.
7. Create the WebSphere variables, the JDBC provider and data source.
8. Enable server to server trust. See "Enabling server-to-server trust" on page 151.
9. Install any subsequent fix packs. The WebSphere Application Server and Jazz™ for Service Management application server versions must have the same release level, including any fix packs. Fixes and upgrades for the run time must be applied manually at each node.
10. Verify the configuration. See "Verifying a successful High Availability configuration" on page 153.

- Update the WebSphere Application Server services with the new Administrative user, specifying the new LDAP user ID as the *WAS_user* and the new LDAP password as the *WAS_user_password*. For more information about updating WebSphere Application Server services, see "updateWasService" on page 453.
- A front-end Network Dispatcher (for example, IBM HTTP Server) must be set up to handle and distribute all incoming session requests. For more information about this task, see Setting up intermediary services.
- Before joining nodes to a High Availability configuration, make sure that each node uses the same file-based repository user ID, which was assigned the role of *iscadmins*.

Common directory locations

Jazz for Service Management topics use path name variables for paths to common directories, for example, home directories.

Jazz for Service Management home directory

The *JazzSM_HOME* variable describes the location where Jazz for Service Management is installed. This location can be specified during installation. If not specified, the following default locations are used:

- Root user installations:

  On AIX, Linux, System z
  `/opt/IBM/JazzSM`

  On Windows
  `C:\Program Files\IBM\JazzSM`

- Non-root user installations:

  On AIX, Linux, System z
  `/home/`*`nonrootuser_name`*`/IBM/JazzSM`

  On Windows
  `C:\Users\`*`nonrootuser_name`*`\IBM\JazzSM`

Jazz for Service Management profile directory

The *JazzSM_WAS_Profile* variable describes the location of the application server profile that is used for Jazz for Service Management. This location is in the /profile/ subdirectory of the Jazz for Service Management home directory.

- Root user installations:

  On AIX, Linux, System z
  /opt/IBM/JazzSM/profile

  On Windows
  C:\Program Files\IBM\JazzSM\profile

- Non-root user installations:

  On AIX, Linux, System z
  /home/*nonrootuser_name*/IBM/JazzSM\profile

  On Windows
  C:\Users*nonrootuser_name*\IBM\JazzSM\profile

Jazz for Service Management profile name

The *JazzSM_Profile_Name* variable refers to the name assigned to the WebSphere Application Server profile for Jazz for Service Management The default name is JazzSMProfile.

Dashboard Application Services Hub home directory

The *DASH_HOME* variable describes the location whereDashboard Application Services Hub is installed. This location can be specified during installation. If not specified, the following default locations are used:

- Root user installations:

  On AIX, Linux, System z
  /opt/IBM/JazzSM/ui

  On Windows
  C:\Program Files\IBM\JazzSM\ui

- Non-root user installations:

  On AIX, Linux, System z
  /home/*nonrootuser_name*/IBM/JazzSM\ui

  On Windows
  C:\Users*nonrootuser_name*\IBM\JazzSM\ui

For more information, see Jazz for Service Management documentation.

Exporting settings repository to a file

You can export the Dynamic Workload Console settings repository from an existing stand-alone Dynamic Workload Console instance to create a file, in XML format, that can be imported into a High Availability configuration.

About this task

If you are setting up a High Availability configuration among existing nodes, you must first export the settings repository from the stand-alone instance and subsequently import the previously exported data after the High Availability configuration is set up.

Note: If you are joining the server to an existing High Availability configuration, the other nodes must not contain custom data, that is, each node must be a clean installation. When you import the settings repository file from the stand-alone instance it is replicated across all other nodes.

To export the settings from a Dynamic Workload Console, perform the following procedure.

Procedure

1. Log in to the Dynamic Workload Console using TWSWEBUIAdministrator credentials.
2. From the navigation toolbar, click **System Configuration** > **Manage Settings**.
3. In the Manage Settings page, click **Export settings** to save the console settings to an XML file in a directory of your choice.
4. Create a new High Availability configuration using the stand-alone server, or join it to an existing configuration.
5. Import the previously exported data to any node in the High Availability configuration by doing as follows:

   In the Manage Settings page, click **Import settings** and browse to the XML file containing the data you want to import.

Results

Create a new High Availability configuration using the stand-alone Dynamic Workload Console, or join it to an existing configuration. After the High Availability configuration is configured, you can import the data file to one of the nodes.

Setting up a High Availability configuration

You can configure multiple Dynamic Workload Console instances to share a DB2 database as a common repository instead of a local directory in a High Availability configuration of nodes.

Before you begin

Note: Exporting data from a Tivoli Integrated Portal High Availability configuration directly to a Dashboard Application Services Hub High Availability configuration is not supported. You must migrate your Tivoli Integrated Portal environment to a Dashboard Application Services Hub environment and then enable a High Availability configuration in the new environment.

Restriction: To upgrade an existing High Availability configuration from Dashboard Application Services Hub Version 3.1 or later to a High Availability configuration in a Dashboard Application Services Hub Version 3.1.2 environment, you must first complete the steps described in "Upgrading a Dashboard Application Services Hub Version 3.1.1 or earlier cluster" on page 167.

If you are creating a High Availability configuration from an existing Dynamic Workload Console instance that contains custom data, ensure that you have exported its data before you begin to configure it for High Availability. After it is configured, you can import the data to one of the nodes in the new configuration.

Dynamic Workload Console is installed on a machine using the cell name designated for all console nodes within the High Availability configuration. Ensure you have installed and set up a network dispatcher (for example, IBM HTTP Server), DB2, and an LDAP as explained in "Requirements" on page 140.

Note: Dynamic Workload Console server must be configured using DB2 without SSL connection. If you want to configure SSL connection, you can enable it after having successfully enabled the High Availability configuration. For more information, see "Enabling SSL for Dashboard Application Services HubServer" on page 170.

About this task

To implement a High Availability configuration, follow the procedure:

Procedure

1. Ensure you have already created a DB2 database on the computer where DB2 is installed (see Creating databases), and completed the procedure to the change the Dynamic Workload Console settings repository from a local file repository to a settings repository on the database. This database is shared as a common repository for nodes that will be subsequently joined to the High Availability configuration. The database administrator must have the ability to create tables. To create a new database, see Creating databases. To use an existing database, see Changing settings repository.
2. Check that you have the JDBC driver for DB2 on the computer where the Dynamic Workload Console is installed. The JDBC driver must be available at: `JazzSM install_dir/lib/db2`.
3. Configure a data source in the WebSphere Application Server administrative console and for the Java Naming and Directory Interface (JNDI) name use `jdbc/tipds`. For information on configuring a data source in the WebSphere Application Server administrative console, see http://www-01.ibm.com/support/knowledgecenter/SSEQTP_8.5.5/com.ibm.websphere.base.doc/ae/tdat_ccrtpds.html.

   In relation to setting up a data source for a high availability configuration in Dashboard Application Services Hub, note the following points:

   - Create the ${DB2UNIVERSAL_JDBC_DRIVER_PATH} variable in the WebSphere Application Server administrative console.
     a. From the WebSphere Application Server console, expand **Environment** and select **WebSphere variables**.
     b. Click the **DB2UNIVERSAL_JDBC_DRIVER_PATH** variable from the list of variables in the right pane to edit the value.
     c. On the Configuration page, enter the path to the directory that contains the DB2 Universal JDBC Driver in the **Value** field.
     d. Click **OK** to save the changes.
   - When creating the new JDBC provider:
     – Create the JDBC provider and data source in the server scope where Dashboard Application Services Hub is deployed, for example, `cells:JazzSMCell:nodes:JazzSMNode:servers:server1`.

- Select **DB2** as the database type.
- Select **DB2 Universal JDBC Driver Provider** as the provider type.
- Select **Connection pool data source** as the implementation type.
- In **Step 2: Enter database class path information**, provide the directory location for DB2 JAR archive files. For example:
  Class path:

  ```
  ${DB2UNIVERSAL_JDBC_DRIVER_PATH}/db2jcc.jar
  ${UNIVERSAL_JDBC_DRIVER_PATH}/db2jcc_license_cu.jar
  ```

  Example of a directory location for WebSphere Application Server `${DB2UNIVERSAL_JDBC_DRIVER_PATH}` variable:
  `C:\IBM\JazzSM\lib\db2`

- When creating a new data source:
  a. In the WebSphere Integrated Solutions Console, open **Resources** > **JDBC** > **Data sources**.
  b. Under **Scope**, select `cells:JazzSMCell:nodes:JazzSMNode:servers:server1` and click **New**.
  c. In Step 1, enter the data source and JNDI names and then click **Next**:
     - Data source name: `tipds`.
     - JNDI name: `jdbc/tipds`.
  d. In Step 2, select the JDBC provider that you created, for example, `DB2 Universal JDBC Driver Provider` and click **Next**.
  e. In Step 3, enter the specific database properties for the data source:
     - Driver type: `4`
     - Database name is the database created in DB2, for example, `dashdb`
  f. Check the **CMP** check box.
  g. Click Next.
  h. In Step 4, set up any necessary security aliases and then click **Next**. Select the DB2 administrator's authentication alias from the **Component-managed authentication alias** option.
  i. In Step 5, the summary of the new data source is provided. Click **Finish**.
  j. Click **Save** to save the changes.
  - Create the Global J2C authentication alias using a DB2 user ID that has permissions to create and modify database tables:
    a. In the WebSphere Integrated Solutions Console, open **Resources** > **JDBC** > **Data sources**.
    b. From the table, click the data source you just created. Click the link, do not just select the check box.
    c. Under **Related Items**, click **JAAS - J2C authentication data**.
    d. Click **New** and set the fields **Alias**, **User ID** (the database of the master domain manager), and the DB2 **Password**.
    e. Click **Apply** and **Save**.
    f. Again, open **Resources** > **JDBC** > **Data sources** and click your data source name.
    g. In **Component-managed authentication alias**, select the J2C authentication alias.
    h. In **Mapping configuration alias** select DefaultPrincipalMapping.
    i. Click **Apply and Save**.

4. Stop and restart the Jazz for Service Management application server.

a. In the *JazzSM_WAS_Profile*/bin directory, for a server named server1, run the following command:

   On Windows

   ```
   stopServer.bat server1
   ```

   On UNIX

   ```
   stopServer.sh server1
   ```

   Note: You are prompted to provide an administrator username and password.

b. In the *JazzSM_WAS_Profile*/bin directory, for a server named server1, run the following command:

   On Windows

   ```
   startServer.bat server1
   ```

   On UNIX

   ```
   startServer.sh server1
   ```

Results

The High Availability configuration is created and the Dynamic Workload Console node is joined to the configuration as the first node.

What to do next

Next, add (or join) additional nodes to the configuration.

Joining a node to a High Availability configuration

You can configure a Dynamic Workload Console to join an existing High Availability configuration.

Before you begin

1. If you are joining a stand-alone Dynamic Workload Console instance to a High Availability configuration, ensure that you first export all of its data. After you join it to the High Availability configuration, you can then import the previously exported data. Other nodes in the configuration must not contain any custom data and should effectively be new installed instances.
2. Make sure that you have successfully enabled High Availability configuration by following the steps in "Setting up a High Availability configuration" on page 143.
3. Make sure that Dynamic Workload Console is installed on the node using the same cell name that is designated for the configuration.
4. Make sure that all console modules deployed to the High Availability configuration are already deployed on the node that you are joining to the configuration.
5. Deploy any wires or transformations used by the nodes in the High Availability configuration.
6. If the High Availability configuration is using any customization changes in consoleProperties.xml, copy these changes and this file to the same location on the node that you are joining to the configuration.
7. Make sure that the node is configured to the same LDAP with the same user and group definitions as all the other nodes in the High Availability configuration.

About this task

The following parameters are used on the `join` option when a node is added:

- **`-Dusername`** - specify the DB2 administrator's username
- **`-Dpassword`** - specify the DB2 administrator's password

Perform the following procedure to join a node to the first node and for every node you want to add to the configuration:

Procedure

1. Check that you have the JDBC driver for DB2 on the computer where the Dynamic Workload Console is installed. The JDBC driver must be available at: `JazzSM install_dir/lib/db2`.
2. Configure a data source in the WebSphere Application Server administrative console and for the Java Naming and Directory Interface (JNDI) name use `jdbc/tipds`. For information on configuring a data source in the WebSphere Application Server administrative console, see http://www-01.ibm.com/support/knowledgecenter/SSEQTP_8.5.5/com.ibm.websphere.base.doc/ae/tdat_ccrtpds.html.

   In relation to setting up a data source for a high availability configuration in Dashboard Application Services Hub, note the following points:

   - Create the ${DB2UNIVERSAL_JDBC_DRIVER_PATH} variable in the WebSphere Application Server administrative console.
     a. From the WebSphere Application Server console, expand **Environment** and select **WebSphere variables**.
     b. Click the **DB2UNIVERSAL_JDBC_DRIVER_PATH** variable from the list of variables in the right pane to edit the value.
     c. On the Configuration page, enter the path to the directory that contains the DB2 Universal JDBC Driver in the **Value** field.
     d. Click **OK** to save the changes.
   - When creating the new JDBC provider:
     – Create the JDBC provider and data source in the server scope where Dashboard Application Services Hub is deployed, for example, `cells:JazzSMCell:nodes:JazzSMNode:servers:server1`.
     – Select **DB2** as the database type.
     – Select **DB2 Universal JDBC Driver Provider** as the provider type.
     – Select **Connection pool data source** as the implementation type.
     – In **Step 2: Enter database class path information**, provide the directory location for DB2 JAR archive files. For example:
       Class path:
       ```
       ${DB2UNIVERSAL_JDBC_DRIVER_PATH}/db2jcc.jar
       ${UNIVERSAL_JDBC_DRIVER_PATH}/db2jcc_license_cu.jar
       ```

       Example of a directory location for WebSphere Application Server ${DB2UNIVERSAL_JDBC_DRIVER_PATH} variable:
       `C:\IBM\JazzSM\lib\db2`
   - When creating a new data source:
     a. In the WebSphere Integrated Solutions Console, open **Resources** > **JDBC** > **Data sources**.
     b. Under **Scope**, select `cells:JazzSMCell:nodes:JazzSMNode:servers:server1` and click **New**.

c. In Step 1, enter the data source and JNDI names and then click **Next**:
   - Data source name: `tipds`.
   - JNDI name: `jdbc/tipds`.

d. In Step 2, select the JDBC provider that you created, for example, `DB2 Universal JDBC Driver Provider` and click **Next**.

e. In Step 3, enter the specific database properties for the data source:
   - Driver type: `4`
   - Database name is the database created in DB2, for example, `dashdb`

f. Check the **CMP** check box.

g. Click Next.

h. In Step 4, set up any necessary security aliases and then click **Next**. Select the DB2 administrator's authentication alias from the **Component-managed authentication alias** option.

i. In Step 5, the summary of the new data source is provided. Click **Finish**.

j. Click **Save** to save the changes.

– Create the Global J2C authentication alias using a DB2 user ID that has permissions to create and modify database tables:

a. In the WebSphere Integrated Solutions Console, open **Resources** > **JDBC** > **Data sources**.

b. From the table, click the data source you just created. Click the link, do not just select the check box.

c. Under **Related Items**, click **JAAS - J2C authentication data**.

d. Click **New** and set the fields **Alias**, **User ID** (the database of the master domain manager), and the DB2 **Password**.

e. Click **Apply** and **Save**.

f. Again, open **Resources** > **JDBC** > **Data sources** and click your data source name.

g. In **Component-managed authentication alias**, select the J2C authentication alias.

h. In **Mapping configuration alias** select DefaultPrincipalMapping.

i. Click **Apply and Save**.

3. Stop and restart the server. For example, for a console server named `server1`, follow these steps:

a. In the *JazzSM_WAS_Profile*`/bin` directory, for a server named `server1`, run the following command:

On Windows

```
stopServer.bat server1
```

On UNIX

```
stopServer.sh server1
```

Note: You are prompted to provide an administrator username and password.

b. In the *JazzSM_WAS_Profile*`/bin` directory, for a server named `server1`, run the following command:

On Windows

```
startServer.bat server1
```

On UNIX

```
startServer.sh server1
```

Results

As a result, the console node is joined to the High Availability configuration.

What to do next

Next, add another node to the High Availability configuration, or if you have completed adding nodes, enable server-to-server trust to every other node in the configuration.

Depending on the Network Dispatcher that you use (for example, IBM HTTP Server) , you might have further updates to get session requests routed to the new node. For more information, refer to the documentation that is applicable to your Network Dispatcher.

Exporting data from a stand-alone server to prepare for High Availability

You can export data from an existing stand-alone Jazz for Service Management application server instance to create a data file that can be imported into a High Availability configuration.

About this task

If you want to add a node to an existing High Availability configuration and the new node contains custom data, you must first export the data before you join the node to the High Availability configuration. The exported data is later imported to one of the nodes in the cluster so that it is replicated across the other nodes in the High Availability configuration.

Procedure

1. At the command line, change to the following directory: *DASH_HOME*/bin/
2. Run the following command to export the stand-alone server data:

   On UNIX and Linux:

   ```
   consolecli.sh Export -username user_name -password password
   -exportFile destination
   ```

   On Windows:

   ```
   .\consolecli.bat Export -username user_name -password password
   -exportFile destination
   ```

   Where:

   user_name
   Specifies the administrator user ID.

   password
   Specifies the password associated with the administrator user ID.

   destination
   Specifies the path and file name for the exported data, for example, c:/tmp/data.zip. If you omit the **-exportFile** *destination* parameter, the exported data is saved to *DASH_HOME*/ui/output/data.zip.
3. Add the node to the High Availability configuration.
4. Import the previously exported data to any node in the High Availability configuration.

Importing stand-alone instance data to a High Availability configuration:

If you exported custom data from a node before joining the node to a High Availability configuration, you can then import the data to any node in the High Availability configuration for it to be replicated across the configuration.

About this task

Import the previously exported data file to any node in the High Availability configuration.

Procedure

1. At the command line, change to the following directory:
   DASH_HOME/bin/
2. On one of the nodes in the High Availability configuration, run the following command to import the data file:
   - consolecli.sh import -username *console_admin_user_ID* -password *console_admin_password* -source *destination*
   - .\consolecli.bat Import -username *user_name* -password *password* -importFile *destination*

   Where:

   user_name
   Specifies the administrator user ID.

   password
   Specifies the password associated with the administrator user ID.

   destination
   Specifies the path and file name to the data file that is to be imported, for example, c:/tmp/data.zip.

Results

The data from the initial Jazz for Service Management application server is imported to the node and replicated across the other nodes in the High Availability configuration.

Configure the Dynamic Workload Console in LDAP

About this task

To configure the Dynamic Workload Console in Lightweight Directory Access Protocol (LDAP), the LDAP must be installed and configured as the user repository for each node in the High Availability configuration. Each node in the High Availability configuration must be enabled to use the same LDAP using the same user and group configuration.

Procedure

1. From the wastools directory, <*DWC_INST_DIR*>/wastools, stop the WebSphere Application Server by issuing the **stopWas.sh** script:
   ./stopWas.sh stopWas.sh -direct -user <username> -password <password>
2. From the wastools directory, back up the WebSphere Application Server configuration by issuing the **backupConfig** command as follows:
   ./backupConfig.sh

3. From the directory, `../LDAPtest/85x`, or the directory, `../LDAPtest/86_9x`, select one of the text files, for example, `EMEA_any2uid.txt`, submit the following from the `wastools` directory:

   ```
   ./changeSecurityProperties.sh /tws_images/LDAP/LDAPtest/86_9x
   /DWC/EMEA_any2uid.txt
   ```

4. From the `wastools` directory, start the WebSphere Application Server by issuing the **startWas.sh** script as follows:

   ```
   ./startWas.sh
   ```

5. Connect to the Dynamic Workload Console with the original user.
6. Go to USER ROLES, select the LDAP user and then select the role.
7. Save your changes.
8. Log in to the Dynamic Workload Console.

Enabling server-to-server trust

Use this procedure to enable nodes to connect to each other and send notifications in High Availability configuration.

About this task

These steps are required to enable High Availability configuration between the participating nodes. Complete these steps on each node.

Procedure

1. In a text editor, open the `ssl.client.props` file from the *JazzSM_profile_dir*`/properties` directory. The default path for the *JazzSM_profile_dir* is `/opt/IBM/JazzSM/profile`.
2. Uncomment the section that starts with **com.ibm.ssl.alias=AnotherSSLSettings** so that it looks like this:

   ```
   com.ibm.ssl.alias=AnotherSSLSettings
   com.ibm.ssl.protocol=SSL_TLS
   com.ibm.ssl.securityLevel=HIGH
   com.ibm.ssl.trustManager=IbmX509
   com.ibm.ssl.keyManager=IbmX509
   com.ibm.ssl.contextProvider=IBMJSSE2
   com.ibm.ssl.enableSignerExchangePrompt=true
   #com.ibm.ssl.keyStoreClientAlias=default
   #com.ibm.ssl.customTrustManagers=
   #com.ibm.ssl.customKeyManager=
   #com.ibm.ssl.dynamicSelectionInfo=
   #com.ibm.ssl.enabledCipherSuites=
   ```

3. Uncomment and modify the section that starts with **com.ibm.ssl.trustStoreName=AnotherTrustStore** to have it look like this:

   ```
   com.ibm.ssl.trustStoreName=AnotherTrustStore
   com.ibm.ssl.trustStore=${user.root}/etc/trust.p12
   com.ibm.ssl.trustStorePassword=trustStore_password
   com.ibm.ssl.trustStoreType=PKCS12
   com.ibm.ssl.trustStoreProvider=IBMJCE
   com.ibm.ssl.trustStoreFileBased=true
   com.ibm.ssl.trustStoreReadOnly=false
   ```

   where, by default the trustStore password is `WebAS`. **Example**:

   ```
   com.ibm.ssl.trustStore=JazzSM_profile_dir/etc/trust.p12
   com.ibm.ssl.trustStorePassword=WebAS
   com.ibm.ssl.trustStoreType=JKS
   ```

   Note: This is a valid example if default IBM Workload Scheduler certificates have been used. If you then want to encrypt the entered password, run the

encryptProfileProperties script wastool, as described in "Application server - encrypting the profile properties files" on page 452.

4. Update the location of the trust store that the signer should be added to in the `com.ibm.ssl.trustStore` property of `AnotherTrustStore` by replacing the default value **com.ibm.ssl.trustStore=${user.root}/etc/trust.p12** with the correct path for your trust store. Example:

   ```
   com.ibm.ssl.trustStore=${user.root}/config/cells/JazzSMNode01Cell/nodes
   /JazzSMNode01/trust.p12
   ```

   After the update, the section must look like this:

   ```
   com.ibm.ssl.trustStoreName=AnotherTrustStore
   com.ibm.ssl.trustStore=${user.root}/config/cells/JazzSMNode01Cell/nodes
   /JazzSMNode01/trust.p12
   com.ibm.ssl.trustStorePassword=trustStore_password
   com.ibm.ssl.trustStoreType=PKCS12
   com.ibm.ssl.trustStoreProvider=IBMJCE
   com.ibm.ssl.trustStoreFileBased=true
   ```

5. Save your changes to `ssl.client.props`.
6. Stop and restart the Jazz for Service Management application server:
   a. In the *JazzSM_WAS_Profile*`/bin` directory, for a server named `server1`, run the following command:

      On Windows

      ```
      stopServer.bat server1
      ```

      On UNIX

      ```
      stopServer.sh server1
      ```

      Note: You are prompted to provide an administrator username and password.

   b. In the *JazzSM_WAS_Profile*`/bin` directory, for a server named `server1`, run the following command:

      On Windows

      ```
      startServer.bat server1
      ```

      On UNIX

      ```
      startServer.sh server1
      ```

7. Complete all of the steps so far on each node before you continue with the remaining steps.
8. Run the following command on each node for each *myremotehost* (that is, for every node that you want to enable trust with) in the High Availability configuration:

   On Windows:

   ```
   JazzSM_profile_dir\bin\retrieveSigners.bat
   NodeDefaultTrustStore AnotherTrustStore -host myremotehost -port
   remote_SOAP_port
   ```

   On UNIX and Linus:

   ```
   JazzSM_profile_dir/bin/bin/retrieveSigners.sh
   NodeDefaultTrustStore AnotherTrustStore -host myremotehost
   -port remote_SOAP_port
   ```

   where `myremotehost` is the name of the computer to enable trust with; `remote_SOAP_port` is the SOAP connector port number (16313 is the default). If

you have installed with non-default ports, use the showHostProperties utility to check the SOAP port number, as described in "Changing host properties" on page 456.

9. Stop and restart WebSphere Application Server by entering the following commands:
   a. `stopWas.bat -direct -user ldapuser -password ldpapwd` (locate the `stopWas.bat` in `TWA_home\wastools directory`.)
   b. `startWas.bat -direct -user ldapuser -password ldpapwd` (locate the `startWas.bat` in `TWA_home\wastools directory`.)

Example

In this example, High Availability configuration is comprised of two Microsoft Windows nodes named *myserver1* and *myserver2*. The command entered on *myserver1*:

```
retrieveSigners.bat NodeDefaultTrustStore AnotherTrustStore -host myserver2
 -port 16313
```

The command entered on *myserver2*:

```
retrieveSigners.bat NodeDefaultTrustStore AnotherTrustStore -host myserver1
 -port 16313
```

Then, enter Dynamic Workload Console user and password, when prompted.

The following is an example of two nodes on Linux, abc.rome.example.com and xyz.rome.example.com. The command entered on abc.rome.example.com:

```
./retrieveSigners.sh NodeDefaultTrustStore AnotherTrustStore
-host xyz.rome.example.com -port 1631
```

The command entered on xyz.rome.example.com:

```
./retrieveSigners.sh NodeDefaultTrustStore AnotherTrustStore
-host abc.rome.example.com -port 1631
```

Verifying a successful High Availability configuration

Use the information in this topic to verify that your Dynamic Workload Console High Availability configuration is working correctly once you have added all nodes and enabled server-to-server trust.

About this task

This task allows you to confirm the following functions are working correctly:

- The database used for your High Availability configuration is properly created and initialized.
- Every node in the configuration uses the database as its repository instead of its own local file system.
- Server-to-server trust is properly enabled between nodes.

To verify your High Availability configuration:

Procedure

1. Ensure that each Dynamic Workload Console instance on every node is running.
2. In a browser, log into one node, create a new View and save your changes.

3. Log into the remaining nodes and verify that the newly created view is available in each one.

Preparing the HTTP server for high availability

Install the IBM HTTP Server and configure the Web server plug-in for passing requests to the Jazz for Service Management application server that are part of the high availability configuration.

Before you begin

The IBM HTTP Server uses a web server plug-in to forward HTTP requests to the Jazz for Service Management application server. You can configure the HTTP server and the web server plug-in to act as the high availability server, that is, pass requests (HTTP or HTTPS) to one of any number of nodes. The high availability methods that are supported by the plug-in are *round robin* and *random*.

- With a round robin configuration, when a browser connects to the HTTP server, it is directed to one of the configured nodes. When another browser connects, it is directed to a different node.
- With the random setting, each browser is connected randomly to a node. When a connection is established between a browser and a particular node, that connection remains until the user logs out or the browser is closed.

The HTTP server is necessary for directing traffic from browsers to the applications that run in the Dashboard Application Services Hub environment. The server is installed between the console and the Jazz for Service Management application server, and is outside the firewall.

The web server plug-in uses the `plugin-cfg.xml` configuration file to determine whether a request is for the Jazz for Service Management application server.

About this task

Complete this procedure to configure the web server plug-in for obtaining high availability for each node.

Procedure

1. If IBM HTTP Server Version 8.5 is not installed, install it before you proceed. It must be installed where it can be accessed from the Internet or intranet (or both).

   Note: Jazz for Service Management bundles the WebSphere Application Server Version 8.5 Supplements installation media, which contains the installation packages for IBM HTTP Server and the IBM HTTP Server plug-in for IBM WebSphere Application Server. If you do not have the DVDs, you can download the electronic images for Jazz for Service Management from IBM Passport Advantage®. See Downloading Jazz for Service Management.
2. Install IBM HTTP Server ensuring that you include the `IBM HTTP Server Plug-in for IBM WebSphere Application Server` option. For more information, see Installing, updating, rolling back, and uninstalling IBM HTTP Server.
3. Create a CMS-type key database. For more information, see Creating a new key database using the command-line interface.
4. Create a self-signed certificate to allow SSL connections between nodes. For more information, see Creating a self-signed certificate.

5. To enable SSL communications for the IBM HTTP Server, in a text editor, open *HTTP_server_install_dir*/conf/httpd.conf. Locate the line # End of example SSL configuration and add the following lines, ensuring that the KeyFile line references the key database file that was created in step 3 on page 154 and save your changes.

```
LoadModule ibm_ssl_module modules/mod_ibm_ssl.so
Listen 443
<VirtualHost *:443>
SSLEnable
SSLProtocolDisable SSLv2
ErrorLog "C:/Program Files (x86)/IBM/HTTPServer/logs/sslerror.log"
TransferLog "C:/Program Files (x86)/IBM/HTTPServer/logs/sslaccess.log"
KeyFile "C:/Program Files (x86)/IBM/WebSphere/Plugins_1/etc/plugin-key.kdb"
SSLStashfile "C:/Program Files (x86)/IBM/WebSphere/Plugins_1/etc/plugin-key.sth"
</VirtualHost>
SSLDisable
```

   For more information, see the first example at Securing with SSL communications.
6. Restart the IBM HTTP Server, For more information, see Starting and stopping IBM HTTP Server.
7. On the IBM HTTP Server computer, to verify that SSL is enabled ensure that you can access https://localhost.
8. Stop and restart the Jazz for Service Management application server:
   a. In the *JazzSM_WAS_Profile*/bin directory, for a server named server1, run the following command:

      On Windows

      ```
      stopServer.bat server1
      ```

      On UNIX

      ```
      stopServer.sh server1
      ```

      Note: You are prompted to provide an administrator username and password.
   b. In the *JazzSM_WAS_Profile*/bin directory, for a server named server1, run the following command:

      On Windows

      ```
      startServer.bat server1
      ```

      On UNIX

      ```
      startServer.sh server1
      ```
9. Start the HTTP server:
   a. Change to the directory where it is installed.
   b. Run this command: bin/apachectl start Note you must restart the server after you change the plugin-cfg.xml file.

What to do next

Enter the URL for the HTTP Server in a browser http://*HTTP_server_host*/*HTTP_server_port* and it is forwarded to one of the nodes.

Note: The default high availability method is random, whereby each browser is connected randomly to a node.

Setting clone IDs for nodes

Assign a clone ID for all nodes in the cluster.

About this task

Complete this procedure to set clone IDs for all nodes in the cluster. You must carry out these steps on each node.

Procedure

1. In a text editor, open the server.xml file from the *JazzSM_WAS_Profile*/config/cells/*JazzSMNode01Cell*/nodes/*JazzSMNode01*/servers/server1 directory
2. In server.xml, locate the entry <components xmi:type="applicationserver.webcontainer:WebContainer.
3. Within the components element, add the following entry:

   ```
   <properties xmi:id="WebContainer_1183077764084" name="HttpSessionCloneId"
    value="12345" required="false"/>
   ```

   Where:

   value is the clone ID for the node, for example, value="12345". The clone ID must be unique to each node. An example of an updated components element is provided here:

   ```
   <components xmi:type="applicationserver.webcontainer:WebContainer"
   xmi:id="WebContainer_1183077764084" enableServletCaching="false"
   disablePooling="false">
         <stateManagement xmi:id="StateManageable_1183077764087"
   initialState="START"/>
         <services xmi:type="applicationserver.webcontainer:SessionManager"
   xmi:id="SessionManager_1183077764084" enable="true" enableUrlRewriting="false"
   enableCookies="true" enableSSLTracking="false"
   enableProtocolSwitchRewriting="false"
   sessionPersistenceMode="NONE" enableSecurityIntegration="false"
   allowSerializedSessionAccess="false" maxWaitTime="5"
   accessSessionOnTimeout="true">
           <defaultCookieSettings xmi:id="Cookie_1183077764084" domain=""
   maximumAge="-1" secure="false"/>
           <sessionDatabasePersistence
   xmi:id="SessionDatabasePersistence_1183077764084"
   datasourceJNDIName="jdbc/
   Sessions" userId="db2admin" password="{xor}Oz1tPjsyNjE="
   db2RowSize="ROW_SIZE_4KB" tableSpaceName=""/>
           <tuningParams xmi:id="TuningParams_1183077764084"
   usingMultiRowSchema="false" maxInMemorySessionCount="1000"
   allowOverflow="true" scheduleInvalidation="false"
   writeFrequency="TIME_BASED_WRITE" writeInterval="10"
   writeContents="ONLY_UPDATED_ATTRIBUTES" invalidationTimeout="30">
             <invalidationSchedule xmi:id="InvalidationSchedule_1183077764084"
   firstHour="14" secondHour="2"/>
           </tuningParams>
         </services>
      <properties xmi:id="WebContainer_1183077764084" name="HttpSessionCloneId"
   value="12345" required="false"/>
       </components>
   ```

4. Save the changes you made to server.xml.

Generating the plugin-cfg.xml file

Run GenPluginCfg.bat to generate the plugin-cfg.xml file and save it in *JazzSM_WAS_Profile*/config/cells.

About this task

Complete this procedure to generate the plug-cfg.xml file. You must carry out these steps on each node.

Important: You must run `GenPluginCfg.bat` to generate a new `plugin-cfg.xml` file each time that a dashboard application is installed into a load balanced Jazz for Service Management application server environment.

Procedure

1. On a node, change to *JazzSM_WAS_Profile*`/bin/` and run the following command:
   - `GenPluginCfg.bat`
   - `GenPluginCfg.sh`

   This command generates a file that is named `plugin-cfg.xml` and saves it to the *JazzSM_WAS_Profile*`/config/cells` directory.
2. On the IBM HTTP Server, in the following directory, replace the existing `plugin-cfg.xml` with the version generated in step 1:

   HTTP_web_server_install_dir`/plugins/config/webserver1`

   The following steps establish the new **/ibm/*** URI (Uniform Resource Identifier), which is where the plug-in redirects requests:
   a. On the IBM HTTP Server, change to the directory where the web server definition file is (such as, `cd plugins/config/webserver1`).
   b. Open the `plugin-cfg.xml` file in a text editor and edit the file to provide details of your IBM HTTP Server and all Jazz for Service Management application server instances. Refer to the sample content provided to assist you in editing `plugin-cfg.xml`. For more information about the `plugin-cfg.xml` file, see the reference material at: http://www-01.ibm.com/support/knowledgecenter/SSAW57_8.5.5/com.ibm.websphere.nd.doc/ae/rwsv_plugincfg.html.

      Note: Where the provided sample differs from the WebSphere Application Server reference information, the WebSphere Application Server takes precedence.

      HTTP SERVER PATH is the path to where the HTTP server is installed.

      HTTP SERVER PORT is the port for the HTTP server.

      SERVER1 is the fully qualified name of the computer where the Jazz for Service Management application server is installed and started.

      SERVER2 is the fully qualified name of the computer where another Jazz for Service Management application server is installed and started.

      CLONE_ID is the unique clone ID assigned to a particular node (server) in the cluster.
   c. In the `ServerCluster` section, the value for the `keyring` property must be ***HTTP SERVER PATH*** `/plug-ins/etc/plug-in-key.kdb` and the value for the `stashfile` property must be ***HTTP SERVER PATH*** `/plug-ins/etc/plug-in-key.sth`.
   d. Continue to add `Server` entries for any other nodes, following the same pattern. Add an entry under `PrimaryServers` for each additional server.
   e. Add `CloneID` and `LoadBalanceWeight` attributes for every `Server` entry.

      Important: For more information about the web server plug-in workload management policies and to help you determine the appropriate values for the elements `LoadBalance` and `LoadBalanceWeight`, see:
      - IBM WebSphere Application Server Network Deployment V5.0 - Workload Management Policies

- Understanding IBM HTTP Server plug-in Load Balancing in a clustered environment

Attention: The HTTP and HTTPS port values for all nodes must be the same.
While the following extract is from an IBM HTTP Server Version 7.0 `plugin-cfg.xml`, its contents are still relevant. Some details are changed for IBM HTTP Server Version 8.5. For more information about the format of the `plugin-cfg.xml` file in IBM HTTP Server Version 8.5, see:plugin-cfg.xml file

```
<Config ASDisableNagle="false" IISDisableNagle="false"
IgnoreDNSFailures="false" RefreshInterval="60"
ResponseChunkSize="64" AcceptAllContent="false"
IISPluginPriority="High" FIPSEnable="false"
AppServerPortPreference="HostHeader" VHostMatchingCompat="false"
ChunkedResponse="false">
 <Log LogLevel="Trace" Name="HTTP SERVER PATH/Plugins/logs/webserver1/
http_plugin.log"/>
 <Property Name="ESIEnable" Value="true" />
 <Property Name="ESIMaxCacheSize" Value="1024" />
 <Property Name="ESIInvalidationMonitor" Value="false" />
 <Property Name="ESIEnableToPassCookies" Value="false" />
 <Property Name="PluginInstallRoot" Value="HTTP SERVER PATH/Plugins" />
 <VirtualHostGroup Name="default_host">
  <VirtualHost Name="*:16310" />
  <VirtualHost Name="*:80" />
  <VirtualHost Name="*:16311" />
  <VirtualHost Name="*:5060" />
  <VirtualHost Name="*:5061" />
  <VirtualHost Name="*:443" />
   <VirtualHost Name="*:HTTP SERVER PORT"/>
 </VirtualHostGroup>
 <ServerCluster CloneSeparatorChange="false" GetDWLMTable="false"
IgnoreAffinityRequests="true" LoadBalance="Round Robin"
Name="server1_Cluster" PostBufferSize="64" PostSizeLimit="-1"
RemoveSpecialHeaders="true" RetryInterval="60">
 <Server Name="TIPNode1_server1"
ConnectTimeout="0" CloneID="CLONE_ID" ExtendedHandshake="false"
ServerIOTimeout="0" LoadBalanceWeight="100" MaxConnections="-1"
WaitForContinue="false">
  <Transport Hostname="SERVER1" Port="16310"
Protocol="http"/>
  <Transport Hostname="SERVER1" Port="16311"
Protocol="https">
   <Property name="keyring" value="HTTP SERVER PATH\Plugins\config
\webserver1\plugin-key.kdb"/>
   <Property name="stashfile" value="HTTP SERVER PATH\Plugins\config
\webserver1\plugin-key.sth"/>
  </Transport>
  </Server>
<Server Name="TIPNode1_server2"
ConnectTimeout="0" CloneID="CLONE_ID" ExtendedHandshake="false"
ServerIOTimeout="0" LoadBalanceWeight="100" MaxConnections="-1"
WaitForContinue="false">
    <Transport Hostname="SERVER2" Port="16310"
Protocol="http"/>
    <Transport Hostname="SERVER2" Port="16311"
Protocol="https">
     <Property name="keyring" value="HTTP SERVER PATH\Plugins\config
\webserver1\plugin-key.kdb"/>
     <Property name="stashfile" value="HTTP SERVER PATH\Plugins\config
\webserver1\plugin-key.sth"/>
    </Transport>
    </Server>
  <PrimaryServers>
      <Server Name="TIPNode1_server1" />
```

```
   <Server Name="TIPNode1_server2" />
  </PrimaryServers>
 </ServerCluster>
 <UriGroup Name="server1_Cluster_URIs">
  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid"
Name="/ivt/*" />
  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid"
Name="/IBM_WS_SYS_RESPONSESERVLET/*" />
  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid"
Name="/HCL_WS_SYS_RESPONSESERVLET/*.jsp" />
  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid"
Name="/HCL_WS_SYS_RESPONSESERVLET/*.jsv" />
  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid"
Name="/HCL_WS_SYS_RESPONSESERVLET/*.jsw" />
  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid"
Name="/HCL_WS_SYS_RESPONSESERVLET/j_security_check" />
  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid"
Name="/HCL_WS_SYS_RESPONSESERVLET/ibm_security_logout" />
  <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/ibm/console/*" />
  <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/ibm/help/*" />
  <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/ibm/action/*" />
  <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/ISCWire/*" />
  <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/isc/*" />
  <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/ISCHA/*" />
  <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/tip_ISCAdminPortlet/*" />
  <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/ISCAdminPortlets/*" />
  <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/mum/*" />
  <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/ibm/TIPChangePasswd/*" />
  <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/ibm/TIPExportImport/*" />
  <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/ibm/tivoli/*" />
  <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/proxy/*" />
  <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/TIPWebWidget/*" />
  <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/ibm/dbfile/*" />
  <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/ibm/TIPChartPortlet/*" />
  <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/TIPUtilPortlets/*" />
  <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/WIMPortlet/*" />
  <Uri AffinityCookie="JSESSIONID_ibm_console_16310"
AffinityURLIdentifier="jsessionid" Name="/SysMgmtCommonTaskGroups/*" />
  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid"
Name="/ibm/tivoli/*"/>
  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid"
Name="/ibm/console/*"/>
  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid"
Name="/ITMOSD/*"/>
 </UriGroup>
 <Route ServerCluster="server1_Cluster" UriGroup="server1_Cluster_URIs"
VirtualHostGroup="default_host" />
 <RequestMetrics armEnabled="false" newBehavior="false" rmEnabled="false"
traceLevel="HOPS">
```

```
  <filters enable="false" type="URI">
   <filterValues enable="false" value="/snoop" />
   <filterValues enable="false" value="/hitcount" />
  </filters>
   <filters enable="false" type="SOURCE_IP">
   <filterValues enable="false" value="255.255.255.255" />
   <filterValues enable="false" value="254.254.254.254" />
  </filters>
  <filters enable="false" type="JMS">
   <filterValues enable="false" value="destination=aaa" />
  </filters>
  <filters enable="false" type="WEB_SERVICES">
   <filterValues enable="false" value="wsdlPort=aaa:op=bbb:nameSpace=ccc" />
  </filters>
 </RequestMetrics>
</Config>
```

Configuring SSL from each node to the IBM HTTP Server

For load balanced implementations, you must configure SSL between the IBM HTTP Server plug-in and each node in the cluster.

Before you begin

IBM HTTP Server is installed and configured for load balancing.

About this task

For each node in the cluster, follow these instructions to configure the node to communicate over a secure (SSL) channel with the IBM HTTP Server.

Procedure

1. Log in to the Dashboard Application Services Hub.
2. In the navigation pane, click **Console Settings** > **Websphere Administrative Console** and click **Launch Websphere administrative console**.
3. Follow these steps to extract signer certificate from the truststore:
   a. In the WebSphere Application Server administrative console navigation pane, click **Security** > **SSL certificate and key management**.
   b. In the Related Items area, click the **Key stores and certificates** link and in the table click the **NodeDefaultTrustStore** link.
   c. In the Additional Properties area, click the **Signer certificates** link and in the table that is displayed, select the `root` entry check box.
   d. Click **Extract** and in the page that is displayed, in the **File name** field, enter a certificate file name (*certificate*`.arm`. For example, `c:\tivpc064ha1.arm`.
   e. From the **Data Type** list, select the **Base64-encoded ASCII data** option and click **OK**.
   f. Locate the extracted signer certificate and copy it to the computer that is running the IBM HTTP Server.

   Note: These steps are particular to Dashboard Application Services Hub, for general WebSphere Application Server details and further information, see: Adding the correct SSL Signer certificates to the plug-in keystore
4. On the computer that is running the IBM HTTP Server, follow these steps to import the extracted signer certificate into the key database:
   a. Start the key management utility (iKeyman), if it is not already running, from *HTTP_SERVER_PATH*`/bin`:
      • At the command line, enter `./ikeyman.sh`

- At the command prompt, enter `ikeyman.exe`

b. Open the CMS key database file that is specified in `plugin-cfg.xml`. For example, *HTTP_SERVER_PATH*`/plug-ins/etc/plug-in-key.kdb`.

c. Provide the password (default is `WebAS`) for the key database and click **OK**.

d. From the **Key database content**, select **Signer Certificates**.

e. Click **Add** and select the signer certificate that you copied from the node to the computer that is running the IBM HTTP Server and click **OK**.

f. Select the **Stash password to a file** check box and click **OK** to save the key database file.

Note: For more information about certificates in WebSphere Application Server, see Receiving a signed certificate from a certificate authority.

5. Repeat these steps for each node in the cluster.
6. For the changes to take effect, stop and restart all nodes in the cluster and also restart the computer that is running the IBM HTTP Server.

a. In the *JazzSM_WAS_Profile*`/bin` directory, for a server named `server1`, run the following command:

On Windows

```
stopServer.bat server1
```

On UNIX

```
stopServer.sh server1
```

Note: You are prompted to provide an administrator username and password.

b. In the *JazzSM_WAS_Profile*`/bin` directory, for a server named `server1`, run the following command:

On Windows

```
startServer.bat server1
```

On UNIX

```
startServer.sh server1
```

c. Restart the IBM HTTP Server. For more information, see Starting and stopping IBM HTTP Server.

What to do next

You can access the load balanced cluster through `https://`*http_server_hostname*`/ibm/console` (assuming that the default context root (`/ibm/console`) was defined in at the time of installation.

Maintaining a high availability cluster

If synchronized data fails to be committed to a node in the cluster, remove that node from the cluster for corrective action. Use the high availability `consolecli` commands to analyze and update cluster nodes.

The following `consolecli.sh|bat` commands for maintaining the cluster are available from *DASH_HOME*`/bin`:

- Run the `ListHAModules` command to return a list of the component modules in the cluster:
  - `consolecli.sh ListHAModules --username` *console_admin_user_ID* `--password` *console_admin_password* `[--nodename true|false]`

– `consolecli.sh ListHAModules --username console_admin_user_ID --password console_admin_password [--nodename true|false]`

`nodename` is optional parameter to the **`ListHAModules`** command. When set to `true`, the local component modules are also listed, otherwise only modules from the database are listed.

- Run the **`ListHANodes`** command to list the current nodes in the cluster, determine whether they are active or not, view their synchronization status, and their version level of Dashboard Application Services Hub:
  – `consolecli.sh ListHANodes --username console_admin_user_ID --password console_admin_password`
  – `consolecli.sh ListHANodes --username console_admin_user_ID --password console_admin_password`
- Run the **`ForceHARefresh`** command on a node to refresh the node with the latest content from the database. The **`ForceHARefresh`** exports data from the database and imports it locally. The database module version for Dashboard Application Services Hub must be lower than local node for export and import to succeed. Use this command to upgrade a cluster when there are no nodes that are in synch with the database module versions. Use the **`ForceHARefresh`** command, verify the result, and then use the **`ForceHAUpdate`** command to update the database.
  – `consolecli.sh ForceHARefresh --username console_admin_user_ID --password console_admin_password`
  – `consolecli.sh ForceHARefresh --username console_admin_user_ID --password console_admin_password`
- To force a database update after running the **`ForceHARefresh`** command, as an administrator, run the **`ForceHAUpdate`** command. The **`ForceHAUpdate`** command pushes the local configuration to the database and updates the modules table to match the local node's module versions. Notifications are sent to other nodes to synchronize. Notified nodes with module versions that match those of the originating node are synchronized. Notified nodes with module versions that do not match, go into maintenance mode until an administrator updates their modules accordingly.
  – `consolecli.sh ForceHAUpdate --username console_admin_user_ID --password console_admin_password`
  – `consolecli.sh ForceHAUpdate --username console_admin_user_ID --password console_admin_password`
- Run the `RemoveHANode` command on a node to remove it from the cluster.

  This command is used to permanently remove a node from the cluster before you delete the WebSphere Application Server data source. If the data source was deleted beforehand, this command can be run from another node to remove a separate node by specifying the relevant node name. There is also an optional `active` parameter that is used for cleanup purposes. Its options are `true|false|unreachable`. If `true` is specified, then all the active nodes in the database with status of active that are not reachable are deleted. If `false` is specified, then all the inactive nodes in the database are deleted. If `unreachable` is specified, then all the nodes that are unreachable from that node are deleted.

  Note: If the `active` parameter is specified, then you do not need to use the `nodename` parameter. If you specify both a `nodename` parameter and the `active` parameter then the `active` parameter is ignored.

  – `consolecli.sh RemoveHANode --username console_admin_user_ID --password console_admin_password [--nodename node_name]|[-- active true|false|unreachable]`

– `consolecli.sh RemoveHANode --username` *`console_admin_user_ID`* `--password` *`console_admin_password`* `[--nodename` *`node_name`*`]|[-- active true|false|unreachable]`

Disabling a node without removing it from the cluster

To disable high availability on a node, in the WebSphere Application Server administrative console, set the value for `com.ibm.isc.ha` custom property to `false`.

Procedure

1. Log in to the Dashboard Application Services Hub.
2. In the navigation pane, click **Console Settings** > **Websphere Administrative Console** and click **Launch Websphere administrative console**.
3. In the WebSphere Application Server administrative console navigation pane, click **Servers** > **WebSphere application servers**.
4. In the **Application Servers** panel, click the console server name link.
5. In the **Configuration** tab, expand the **Java and Process Management** list in the **Server Infrastructure** section.
6. Select the **Process definition** link and in the page that is displayed, click the **Java Virtual Machine** link.
7. In the page that is displayed, click the **Custom properties** link.
8. In the **Custom properties** page, select the **com.ibm.isc.ha** link to display its properties page; or if the property is not listed, click **New** and create a property named `com.ibm.isc.ha`.
9. Set the `com.ibm.isc.ha` property value to `false` and click **OK** to save your settings.
10. Close the WebSphere Application Server administrative console.
11. Stop and restart the server.
    a. In the *`JazzSM_WAS_Profile`*`/bin` directory, for a server named `server1`, run the following command:

       On Windows

       ```
       stopServer.bat server1
       ```

       On UNIX

       ```
       stopServer.sh server1
       ```

       Note: You are prompted to provide an administrator username and password.

    b. In the *`JazzSM_WAS_Profile`*`/bin` directory, for a server named `server1`, run the following command:

       On Windows

       ```
       startServer.bat server1
       ```

       On UNIX

       ```
       startServer.sh server1
       ```

Results

The node is disabled. You can subsequently delete the `com.ibm.isc.ha` property to return the node to the high availability cluster.

Permanently removing a single node

Use the RemoveHANode command to permanently remove a node from a high availability cluster.

About this task

You can use the steps described to remove a node from a Dashboard Application Services Hub high availability cluster or you can remove all nodes by running the RemoveHANode command on each node.

Procedure

1. Stop and restart the server.
   a. In the *JazzSM_WAS_Profile*/bin directory, for a server named server1, run the following command:

      On Windows

      ```
      stopServer.bat server1
      ```

      On UNIX

      ```
      stopServer.sh server1
      ```

      Note: You are prompted to provide an administrator username and password.

   b. In the *JazzSM_WAS_Profile*/bin directory, for a server named server1, run the following command:

      On Windows

      ```
      startServer.bat server1
      ```

      On UNIX

      ```
      startServer.sh server1
      ```

2. To remove the last node from a cluster, see step 3 on page 165. For any other node, remove the WebSphere Application Server data source that was created when the node was joined to the cluster, stop and restart the server, and on a separate node, do the following:
   a. At the command line, change to the following directory:

      DASH_HOME/bin/

   b. Run the ListHANodes command to list the current cluster nodes and their statuses:

      On Windows

      ```
      consolecli.bat ListHANodes --username console_admin_user_ID
      --password console_admin_password
      ```

      On UNIX

      ```
      consolecli.sh ListHANodes --username console_admin_user_ID
      --password console_admin_password
      ```

      Where:

      console_adsmin_user_ID
      Specifies the console administrator user ID.

      console_admin_password
      Specifies the password associated with the administrator user ID.

   c. Run the following command to remove the specified node:

On Windows

```
consolecli.bat RemoveHANode --username console_admin_user_ID
--password console_admin_password --nodename node_name
```

On UNIX

```
consolecli.sh RemoveHANode --username console_admin_user_ID
--password console_admin_password --nodename node_name
```

Where:

console_adsmin_user_ID
Specifies the console administrator user ID.

console_admin_password
Specifies the password associated with the administrator user ID.

node_name
Specifies the node that you want to remove. Refer to the node names returned by the `ListHANodes` command in step 2b on page 164 to ensure that you have the correct node name.

3. For the last node in the cluster, run the `RemoveHANode` command specifying its node name, as described in step 2c on page 164, and then remove the WebSphere Application Server data source that was created when the node was joined to the cluster.
4. In each case, stop and restart the server.

Results

The relevant node is removed from the high availability cluster and is now a stand-alone console.

Permanently removing nodes by activity status

Use the `RemoveHANode` command's `--active` parameter to permanently remove nodes from a cluster, based on their activity status.

About this task

Use the optional `--active` parameter with the **`RemoveHANode`** command, to remove nodes that are active, inactive, or unreachable. The `--active` parameter is useful for cleanup purposes. Its options are `true`, `false`, and `unreachable`.

Procedure

1. Stop and restart the server.
   a. In the *JazzSM_WAS_Profile*`/bin` directory, for a server named `server1`, run the following command:

      On Windows

      ```
      stopServer.bat server1
      ```

      On UNIX

      ```
      stopServer.sh server1
      ```

      Note: You are prompted to provide an administrator username and password.
   b. In the *JazzSM_WAS_Profile*`/bin` directory, for a server named `server1`, run the following command:

On Windows

```
startServer.bat server1
```

On UNIX

```
startServer.sh server1
```

2. At the command line, on one node, change to the following directory:
   DASH_HOME/bin/
3. Optional: Run the ListHANodes command to list the current cluster nodes and their statuses:
   - consolecli.sh ListHANodes --username *console_admin_user_ID* --password *console_admin_password*
   - consolecli.sh ListHANodes --username *console_admin_user_ID* --password *console_admin_password*

   Where:

   console_adsmin_user_ID
   Specifies the console administrator user ID.

   console_admin_password
   Specifies the password associated with the administrator user ID.
4. Run the following command, one or more times, specifying the relevant --active option, to remove some or all of the listed nodes, based on their activity status:
   - consolecli.sh RemoveHANode --username *console_admin_user_ID* --password *console_admin_password* --active true|false|unreachable
   - consolecli.sh RemoveHANode --username *console_admin_user_ID* --password *console_admin_password* --active true|false|unreachable

   Where:

   console_adsmin_user_ID
   Specifies the console administrator user ID.

   console_admin_password
   Specifies the password associated with the administrator user ID.

   --active true | false | unreachable
   Specifies whether you want to remove active, inactive, or unreachable nodes. If true is specified, then all the active nodes in the database with status of active are deleted. If false is specified, then all the inactive nodes in the database are deleted. If unreachable is specified, then all the nodes that are unreachable from that node are deleted. Refer to the nodes returned by the ListHANodes command in 2 to ensure that you know the activity status of each node in the cluster.
5. Remove the WebSphere Application Server data source that was created when the nodes were joined to the cluster.
6. Stop and restart the server.

Results

The nodes that have an activity status matching the option that you specified for the --active parameter are removed from the load balanced cluster.

Upgrading an existing High Availability configuration

You can upgrade an existing High Availability configuration starting from Dashboard Application Services Hub, version 3.1.2 or earlier to Dashboard Application Services Hub, version 3.1.2.1 environment .

The procedure for upgrading a Dashboard Application Services Hub load-balanced cluster in a Jazz for Service Management environment differs based on the Dashboard Application Services Hub version you have installed:

- For information about upgrading a Dashboard Application Services Hub Version 3.1.1 or earlier cluster, see "Upgrading a Dashboard Application Services Hub Version 3.1.1 or earlier cluster.".
- For information about upgrading a Dashboard Application Services Hub Version 3.1.2 or later cluster, see "Upgrading a Dashboard Application Services Hub Version 3.1.2 cluster" on page 168.

Upgrading a Dashboard Application Services Hub Version 3.1.1 or earlier cluster

Upgrade an existing High Availability configuration from Dashboard Application Services Hub Version 3.1.1 or earlier to a High Availability configuration in a Dashboard Application Services Hub, Version 3.1.2.1 environment. After performing the upgrade, you can proceed with the upgrade of the Dynamic Workload Console.

About this task

To move nodes from an existing High Availability configuration, you must remove all nodes from the old configuration and recreate the High Availability configuration in the new environment. On each node in the old configuration, except for the last node, run the **disjoin** command to remove it from the High Availability configuration. On the last node, you must first edit a script file and then run the **uninstall** command to remove the last node from the old configuration and clean up the database. You are then ready to create a High Availability configuration in the new environment.

Procedure

1. In the existing environment for each node, except the last node, from a command prompt, change to the *DASH_HOME*/bin/ha directory and run the following command to disjoin it from the cluster:
   - *JazzSM_WAS_Profile*/bin/ws_ant.sh -f uninstall.ant disjoin -DdeleteExistingDataSource=true -Dusername=*DB2_username* -Dpassword=*DB2password* -DWAS_username=*WAS_admin_username* -DWAS_password=*WAS_admin_password*
   - *JazzSM_WAS_Profile*\bin\ws_ant.bat -f uninstall.ant disjoin -DdeleteExistingDataSource=true -Dusername=*DB2_username* -Dpassword=*DB2password* -DWAS_username=*WAS_admin_username* -DWAS_password=*WAS_admin_password*

   Important: The -DdeleteExistingDataSource parameter is mandatory and must be set to true.
2. On the last remaining node, locate the following file and open it in a text editor:

   DASH_HOME/bin/ha/uninstall.ant
3. Locate the section that begins with <target name="uninstall" and edit that section so that it reads as follows:

```
<target name="uninstall" depends="checkinput,detectCurrentOSFamily
,setOSFileSeparator,resolveOsgiCfgInitExecutable,exportData,serverStatus">
    <java classname="DatastoreUninstall" classpathref="ha.uninstall.classpath"
failonerror="true">
      <arg value="uninstall"/>
            <arg value="${username}"/>
            <arg value="${password}"/>
        </java>
        <delete file="${ProfilePath}/config/cells/${CellName}/applications
/${IscAppName}.ear/deployments/${IscAppName}
/isclite.war/WEB-INF/tipha.properties"/>
        <replace file="tipha.properties" token="HAEnabled=true"
value="HAEnabled=false" />
        <antcall target="deleteDatasource"/>
        <antcall target="importData"/>
     </target>
```

Tip: The string `exportData` must be added to the first line of code and the line `<antcall target="importData"/>` must be added to the end of the extract.

4. Save your changes to *DASH_HOME*/bin/ha/uninstall.ant.
5. On the last remaining node in the old High Availability configuration, from a command prompt, change to the *DASH_HOME*/bin/ha directory and run the following command:
   - *JazzSM_WAS_Profile*/bin/ws_ant.sh -f uninstall.ant uninstall -DdeleteExistingDataSource=true -Dusername=*DB2_username* -Dpassword=*DB2_password*
   - *JazzSM_WAS_Profile*\bin\ws_ant.bat -f uninstall.ant uninstall -DdeleteExistingDataSource=true -Dusername=*DB2_username* -Dpassword=*DB2_password* -DWAS_username=*WAS_admin_username* -DWAS_password=*WAS_admin_password*

   Important: The `-DdeleteExistingDataSource` parameter is mandatory and must be set to `true`.
6. Upgrade each node to Dashboard Application Services Hub Version 3.1.2.1, then proceed with the upgrade of the Dynamic Workload Console. After these two components have been upgraded, you can recreate the High Availability configuration in the new environment, see "Setting up a High Availability configuration" on page 143.

Upgrading a Dashboard Application Services Hub Version 3.1.2 cluster

Upgrade an existing High Availability configuration from Dashboard Application Services Hub Version 3.1.2 to a High Availability configuration in a Dashboard Application Services Hub Version 3.1.2.1 environment. After performing the upgrade, you can proceed with the upgrade of the Dynamic Workload Console.

About this task

When upgrading, first remove all nodes from the cluster, then perform the upgrade procedure, and add the nodes to the cluster again:

Procedure

1. Remove all nodes belonging to the Dashboard Application Services Hub high availability cluster Version 3.1.2 as described in "Permanently removing a single node" on page 164.
2. Upgrade one of the nodes to Dashboard Application Services Hub Version 3.1.2.1.

3. Test the upgraded node to ensure that it is running as expected.
4. Upgrade each of the remaining nodes, ensuring that you confirm that each one is running as expected before continuing to upgrade subsequent nodes.
5. Join all nodes as described in "Joining a node to a High Availability configuration" on page 146.
6. Verify the configuration as described in "Verifying a successful High Availability configuration" on page 153.

Configuring Dynamic Workload Console to use DB2

Configure Dynamic Workload Consoles to use a database as the settings repository, to have all the consoles share the same settings, thus providing high scalability and availability.

Before you begin

Make sure you have configured Dashboard Application Services Hub (Dynamic Workload Console) to work in High Availability mode and that you have the TWSWEBUIAdministrator role.

About this task

To configure Dynamic Workload Consoles to use and share a database as the settings repository, you must:

1. Create a database for the Dynamic Workload Console. You can also use the same database created for Dashboard Application Services Hub 2.2, but it is recommended that you use a different one.
2. Optionally, configure SSL connection between DB2 and the Dynamic Workload Console.
3. If you have configured an SSL connection, you must also make it active on Dashboard Application Services Hub. See: "Enabling SSL for Dashboard Application Services HubServer" on page 170.
4. Set up the connection between the database and the Dashboard Application Services Hub server. See: "Creating datasource" on page 171.
5. Configure all the Dynamic Workload Console instances to share the same settings repository. See "Sharing a settings repository" on page 173.
6. Optionally, if you want the Dynamic Workload Console to access the database repository with a user without database administrator privileges, you must change the user that updates the settings repository on DB2. See:"Changing the Dynamic Workload Console user of DB repository" on page 174 or "Changing the Dashboard Application Services Hub user of DB repository" on page 175.

To create a database for the Dynamic Workload Console, perform the following procedure:

Procedure

1. Open the DB2 control center, right-click **All Databases**, and select **Create Database** > **Standard**.
2. In the Create Database Wizard, type the database name and click **Finish** to accept all the default options.

Configuring DB2 in SSL mode

Set up your DB2 server for SSL support.

Before you begin

Make sure you have logged in as the DB2 instance owner and set the following configuration parameters and the DB2COMM registry variable.

Use the `db2 update dbm cfg using` *`parameter_name parameter_value`* command, where *parameter_name* is the name of the parameter to be set and *parameter_value* is the value of the parameter to be set.

Procedure

1. Set the `ssl_svr_keydb` configuration parameter to the fully qualified path of the key database file. For example; `C:\TWS\installations\tws850cli\TWS\ssl\gskit\TWSClientKeyStore.kdb` where `TWSClientKeyStore.kdb` is the fully-qualified file name of the KeyStore that stores the DB2 certificate and the trusted certificates.

   Note: It must be recognized by the JKS WebSphere Application Server certificate. If `ssl_svr_keydb` is null (unset), SSL support is not enabled.
2. 2. Set `ssl_svr_stash` configuration parameter to the fully qualified path of the stash file. For example: `C:\TWS\installations\tws850cli\TWS\ssl\gskit\TWSClientKeyStore.sth`. If `ssl_svr_stash` is null (unset), SSL support is not enabled.
3. 3. Set `ssl_svr_label` configuration parameter to the label of the digital certificate of the server. If `ssl_svr_label` is not set, the default certificate in the key database is used. If there is no default certificate in the key database, SSL is not enabled. For example: `client`.
4. 4. Set `ssl_svcename` configuration parameter to the port that the DB2 database system uses for SSL connections. If TCP/IP and SSL are both enabled (the DB2COMM registry variable is set to 'TCPIP, SSL'), set `ssl_svcename` to a port different from the one set for `svcename`. The `svcename` configuration parameter sets the port that the DB2 database system uses for TCP/IP connections. If you set `ssl_svcename` to the same port as `svcename`, neither TCP/IP or SSL are enabled. If `ssl_svcename` is null (unset), SSL support is not enabled.

   Note: When the DB2COMM registry variable is set to 'TCPIP,SSL', if TCPIP support is not properly enabled, for example because to the `svcename` configuration parameter set to null, the error SQL5043N is returned and SSL support is not enabled.
5. Add the value SSL to the DB2COMM registry variable. For example: `db2set -i db2inst1 DB2COMM=SSL`. The database manager can support multiple protocols at the same time. For example, to enable both TCP/IP and SSL communication protocol, specify: `db2set -i db2inst1 DB2COMM=SSL,TCPIP` where: `db2inst1` is the DB2 instance name
6. Restart the DB2 instance. For example:

   ```
   db2stop
   db2start
   ```

Enabling SSL for Dashboard Application Services HubServer

Configure Dashboard Application Services Hub Server to use SSL connection.

Before you begin

Make sure you have successfully configured Dashboard Application Services Hub server High Availability without SSL connection.

About this task

To enable SSL for Dashboard Application Services Hub server, perform the following steps:

Procedure

1. From the command line interface, go to {TWA_HOME}\wastools directory, and run the following command:
   - **On Windows systems**

     changeTIPDatasource.bat *datasourceName* useSsl *portNumber*
   - **On UNIX systems**

     ./ changeTIPDatasource.sh *datasourceName* useSsl *portNumber*

   where,

   datasourceName
   Is the JNDI name of the datasource used for Dashboard Application Services Hub High Availability (specified in tipha.properties For example, DBDatasource=jdbc/tipds.

   useSsl Can be true or false. Specify **true** to enable SSL.

   portNumber
   specify the SSL port number (the same value specified in ssl_svcename parameter) **Example**: ./changeTIPDatasource.sh tipds true 60000.
2. Restart WebSphere Application Server.

Creating datasource

Create datasource.

Procedure

1. Edit the TDWCDatasource.properties file to insert the correct values for the connection parameters to the created database. TDWCDatasource.properties is located in: TWA_home\wastools. See the following sample for your reference:

```
################################
# Datasource properties template
################################

# Host name of the server on which the DB2 is installed
databaseServerName=localhost

# Port used by DB2
databasePort=50000

# Name of the database to use (must exist)
databaseName=TDWC

# If true, when a JDBC provider with provided name is found in the
#configuration, it is deleted and re-created.
# type in "true" just the first time you create the datasource
deleteAndRecreate=false


################################
# Optional properties
################################

# Use SSL connection (FIPS mode). If true, secure connection socket is
#used to communicate with DB2 (default false)
#useSslConnection=false
```

```
# JNDI name to associate with datasource (to be specified in the DWC
#configuration) datasourceJndiName=jdbc/TDWC

# Name of the WebSphere JBDC provider to create
#providerName=tdwcDriver

# Name of the datasource to create
#datasourceName=tdwcDatasource

# Name of the WebSphere node on which is running the DWC.
#nodeName=TIPNode


##############################
# Connection pool properties
#
# These properties are optional.
#
# Look at "connection pool" settings on WAS infocenter for more info
# http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/
#com.ibm.webSphere.express.doc/info/exp/ae/udat_conpoolset.html
##############################

#connectionTimeout = 180
#maxConnections = 20
#minConnections=1
#reapTime=180
#unusedTimeout=1800
#agedTimeout=0
#purgePolicy=EntirePool
```

2. From the command line interface, go to `{TWA_HOME}\wastools` directory, and run the following command to create a datasource:
   - **On Windows systems**

     ```
     installTDWCDataSource.bat TDWCDatasource.properties
     ```
   - **On UNIX systems**

     ```
     ./installTDWCDataSource.sh TDWCDatasource.properties
     ```
3. Check that you have the JDBC driver for DB2 on the computer where the Dynamic Workload Console is installed. The JDBC driver must be available at: JazzSM install_dir/lib/db2.
4. Create the ${DB2UNIVERSAL_JDBC_DRIVER_PATH} variable in the WebSphere Application Server administrative console:
   - From the WebSphere Application Server console, expand **Environment** and select **WebSphere variables**.
   - Click the **DB2UNIVERSAL_JDBC_DRIVER_PATH** variable from the list of variables in the right pane to edit the value.
   - On the Configuration page, enter the path to the directory that contains the DB2 Universal JDBC Driver in the Value field.
   - Click **OK** to save the changes.
5. When creating the new JDBC provider:
   - Create the JDBC provider and data source in the server scope where Dashboard Application Services Hub is deployed, for example, cells:JazzSMCell:nodes:JazzSMNode:servers:server1.
   - Select **DB2** as the database type.
   - Select **DB2 Universal JDBC Driver Provider** as the provider type.
   - Select **Connection pool data source** as the implementation type.
   - In **Step 2: Enter database class path information**, provide the directory location for DB2 JAR archive files. For example:

Class path:

```
${DB2UNIVERSAL_JDBC_DRIVER_PATH}/db2jcc.jar
${UNIVERSAL_JDBC_DRIVER_PATH}/db2jcc_license_cu.jar
```

Example of a directory location for WebSphere Application Server ${DB2UNIVERSAL_JDBC_DRIVER_PATH} variable: C:\IBM\JazzSM\lib\db2

6. Restart WebSphere Application Server.

What to do next

Because the `TDWCDatasource.properties` file is a local file, you must edit it and run the above steps on all Dynamic Workload Console instances.

Sharing a settings repository

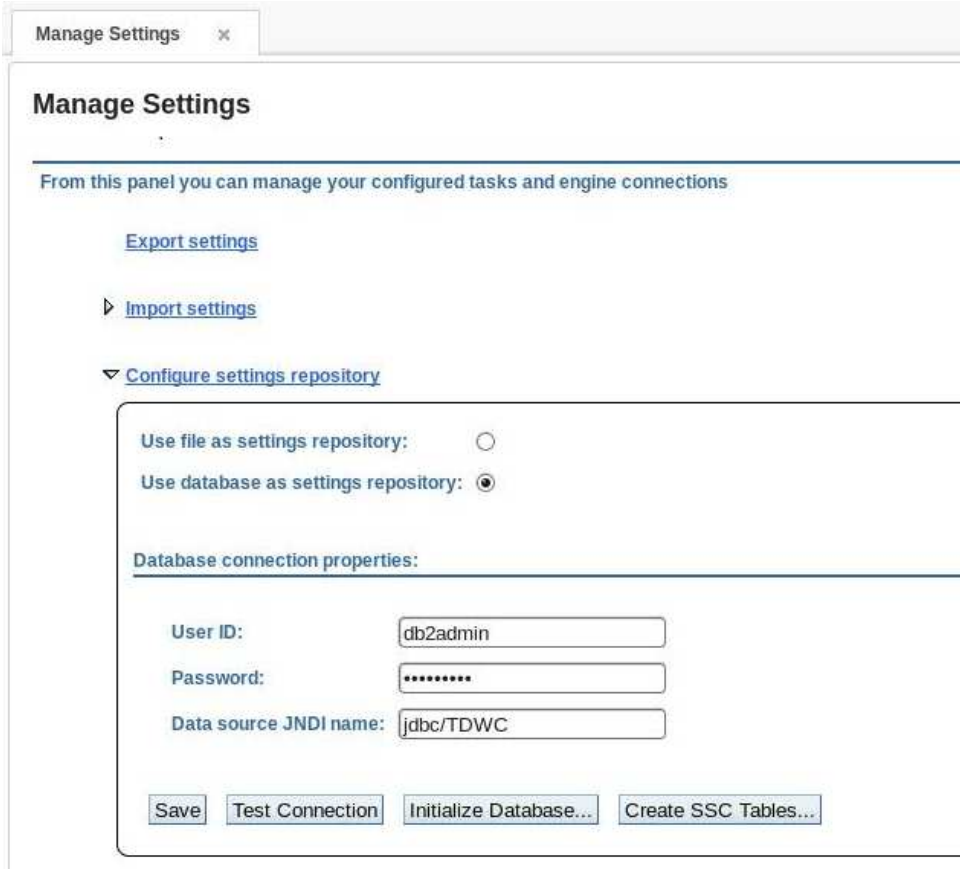
How to share a settings repository on multiple Dynamic Workload Console instances.

Before you begin

To perform this task you must have the TWSWEBUIAdministrator role.

Procedure

1. Ensure that all the Dynamic Workload Console instances that are to share the same settings repository, also use the same user registry.
2. From the Dynamic Workload Console, click **Settings** > **Manage Settings**.
3. In the same panel, click **Configure Settings Repository** > **Use database as settings repository** to specify that settings must be saved in the database instead of a local file.
4. In the **Database Settings** section, specify the credentials required to connect to the database:



5. Optionally, you can test the connection to ensure that you can connect to the database.
6. Save the new configuration to create the `db.properties` file in the *<JazzSM_profile_dir>*`/registry` directory, by default, this is `/opt/IBM/JazzSM/profile/registry`.
7. On the first Dynamic Workload Console that you configure, click **Initialize Database** to initialize it. You can click **Initialize Database** to erase and re-create the database anytime, losing the saved user preferences.

   Note: All the Dynamic Workload Console instances that must be synchronized, must switch to DB2 configuration. If you switch one Dynamic Workload Console, you must also switch all the others.

Results

As a result, all user settings are saved in the database, shared by all the Dynamic Workload Console instances, and all the operations involving user settings are run against this settings repository.

Changing the Dynamic Workload Console user of DB repository

How to change the Dynamic Workload Console user that updates the settings repository on DB2.

Before you begin

To perform this task you need to have the TWSWEBUIAdministrator role.

You must have switched the Dynamic Workload Console settings repository from a local file to a database repository, as described in Changing settings repository.

About this task

Only users with database administrator rights are authorized to initialize the Dynamic Workload Console related tables on the database.

If you want the Dynamic Workload Console to access the database repository with a user without database administrator privileges you must follow these steps:

Procedure

1. Create a new DB2 user and grant this user with SELECT, INSERT, UPDATE, DELETE rights on all the following tables, belonging to TDWC schema:

   ```
   TDWC_EngineConnection
   TDWC_QueryTask
   TDWC_ReportTask
   TDWC_MEQueryTask
   TDWC_Credential
   TDWC_ConfigurationProperty
   TDWC_Preferenceable
   ```

   The above are the default permissions. However, if you need to restrict your policy, you can give the following permissions to the new DB2user:

   ```
   revoke connect,bindadd, createtab, implicit_schema on database from public;
   revoke use of tablespace USERSPACE1     from public;

   grant use of tablespace userspace1 to  user twsdb2;
   grant createtab on database to user  twsdb2;
   grant implicit_schema on database to user twsdb2;
   ```

2. Change Dynamic Workload Console user accessing DB2
   a. From the navigation toolbar, click **System Configuration** > **Manage Settings**.
   b. In the **Database Settings** section, specify the credentials of the newly created user that must to connect to the database.

   Note: As a result of this user switch, theDynamic Workload Console without database administrator privileges will no longer be authorized to the following actions in the Manage Settings panel:

   - **Initialize database**
   - **Import settings** with **Cancel and re-create** option.

Changing the Dashboard Application Services Hub user of DB repository

How to change the Dashboard Application Services Hub user that updates the settings repository on DB2.

Before you begin

You must have switched the Dynamic Workload Console settings repository from a local file to a database repository, as described in the section about changing settings repository in *Dynamic Workload Console User's Guide*.

About this task

If you want the Dashboard Application Services Hub to access the database repository with a user without database administrator privileges you must follow these steps:

Procedure

1. Create a new DB2 user and grant this user with CONNECT, CREATETAB, LOAD rights. For example, `db2 GRANT CONNECT,CREATETAB,LOAD ON DATABASE TO USER db2user2`
2. On each Dashboard Application Services Hub node, configure the high availability system by following the steps in "Setting up a High Availability configuration" on page 143, changing the parameters in `tipha.properties` file with the new information.

   For example, if the new database name is `tipdb2`, you must set the following properties:

   ```
   DBName=tipdb2
   DBDatasource=jdbc/tipds2
   DBDatasourceName=tipds2
   HAEnabled=false
   ```
3. Run the `ws.ant` script specifying the new DB2 user

   For example, `../ws_ant.sh -f install.ant configHA -Dusername=db2user2 -Dpassword=`*`pass`*

Configuring High Availability for multiple IBM Workload Scheduler for z/OS servers

Setting up a High Availability configuration to work with multiple IBM Workload Scheduler for z/OS servers.

Before you begin

You must have configured High Availability as described in "Configuring High Availability for Dynamic Workload Console" on page 138.

About this task

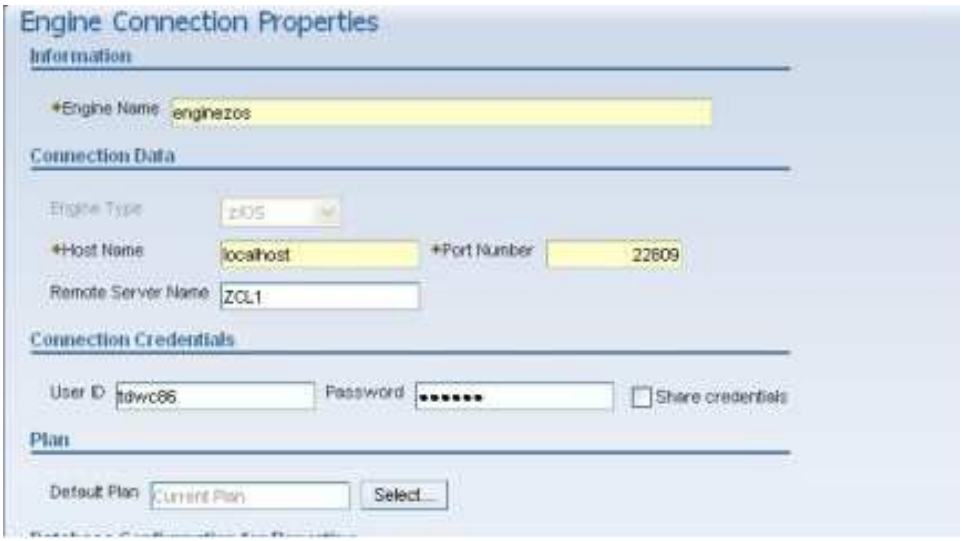
To set up a High Availability in a z/OS environment and distribute the workload across multiple Dynamic Workload Consoles and multiple IBM Workload Scheduler for z/OS servers, perform the following procedure.

Procedure

1. Install a IBM Workload Scheduler for z/OS connector sharing WebSphere Application Server with an already installed Dynamic Workload Console.

   Note: During the installation, use the same ports and the same user names on each node.
2. Create an engine on each IBM Workload Scheduler for z/OS connector, using the same names (for example, ZCL1) and hostnames for all the engines, but specifying different port numbers, to define connections pointing to different IBM Workload Scheduler for z/OS servers for the same controller.
3. Open the `TWA_home\wastools` folder and run the `createZosEngine.sh` (`createZosEnginebat` on Windows) script to create the connection. For example, `./createZosEngine.sh -name ZCL1 -hostName x.xxx.xxx.xx -portNumber 3446`

4. If you want to connect to multiple controllers, repeat this operation using a different engine name to create additional connections. As a result, the same set of engines is defined on each connector, and all the engines use the same name and point to the same controller, through a different server.
5. From one of the Dynamic Workload Console instances, create an engine connection specifying the **Remote Server Name** defined in the IBM Workload Scheduler for z/OS connector (for example, ZCL1), and the **Host Name** as **local host**, to use the IBM Workload Scheduler for z/OS connector locally installed with the Dynamic Workload Console. Define an engine connection for each engine name defined in the step 4.



Results

As a result, the High Availability configuration will route users to different Dynamic Workload Console server, while each Dynamic Workload Console uses the IBM Workload Scheduler for z/OS connector locally installed on the same WebSphere Application Server and a different IBM Workload Scheduler for z/OS server.

Managing Dynamic Workload Console settings repository

User settings like user preferences, saved tasks and engine connections are stored in the settings repository, which by default is a local file. However, you can change this setting and use the database settings repository for all Dynamic Workload Console operations that involve user settings. This can be useful, for example, for scalability purposes or to have multiple Dynamic Workload Console instances sharing the same user settings.

To use a database as your settings repository, you must configure the database settings, as described in the sections about changing and sharing the settings repository, in the *IBM Workload Automation: Dynamic Workload Console User's Guide*.

Configuring Dynamic Workload Console to view reports

This topic describes the configuration steps that you perform to be able to see the reports from the Dynamic Workload Console.

To access the databases where reports are stored, you must have the following prerequisites:

- A user ID and password to access the database
- A working connection between the Dynamic Workload Console and the database

Perform the following steps on the system where the IBM Workload Scheduler engine is running:

- "Configuring for a DB2 database"
- "Configuring for an Oracle database" on page 179

Configuring for a DB2 database

Before you begin

In the case where DB2 on the master domain manager is using the DB2 JDBC Type 2 driver, and the Dynamic Workload Console from where you want to work with reports is on a workstation different from the master domain manager, there are a few configuration steps that you must perform on the Dynamic Workload Console workstation to enable a successful connection to the IBM Workload Scheduler engine.

1. Install the DB2 client on the Dynamic Workload Console workstation.
2. To connect the DB2 client to the server, run the following commands in this order:

   ```
   db2 catalog tcpip node <TWS_NODE_NAME> remote <TWS_HOST> server <TWS_SRVC_PORT>
   db2 attach to <TWS_NODE_NAME> user <TWS_ADMIN_USER> using "<TWS_ADMIN_PW>"
   db2 catalog db <TWS_DB> at node <TWS_NODE_NAME>
   ```

   where,

   <TWS_NODE_NAME>
   The node name, for example, TWS_ND.

   <TWS_HOST>
   The host name of the DB2 server workstation.

   <TWS_SRVC_PORT>
   The port number of the DB2 server workstation.

   <TWS_ADMIN_USER>
   The user name of the DB2 server user.

   <TWS_ADMIN_PW>
   The password of the DB2 server user.

   <TWS_DB>
   The IBM Workload Scheduler database name.

   An example might be:

   ```
   db2 catalog tcpip node TWS_ND remote nc125139.romelab.it.ibm.com server 50000
   db2 attach to TWS_ND user db2admin using "db2admin"
   db2 catalog db TWS at node TWS_ND
   ```

3. Stop the Dynamic Workload Console WebSphere Application Server.
4. Edit the `setupCmdLine.sh` script located in the path *<JAZZSM_Profile>*`/bin/` by adding the following lines at the end of the script:

   ```
   LD_LIBRARY_PATH=$LD_LIBRARY_PATH:<DB2_CLIENT_HOME>/lib64; export LD_LIBRARY_PATH
   LIBPATH=$LIBPATH:<DB2_CLIENT_HOME>/lib64; export LIBPATH SHLIB_PATH=$SHLIB_PATH:
   <DB2_CLIENT_HOME>/lib64; export SHLIB_PATH
   ```

5. Restart the Dynamic Workload Console.

About this task

For DB2, the IT administrator, or the IBM Workload Scheduler IT administrator, or both working together, do the following:

1. Create an operating system user and specify a password.
2. Launch the following script:

   ```
   <TWA_home>/TWS/dbtools/DB2/scripts/dbgrant.bat/.sh
     <ID_of_user_to_be_granted>
     <database_name>
     [<database_admin_user> <password>]
   ```

   where the variables are as follows:

   <TWA_home>
   The IBM Workload Automation instance directory

   <ID_of_user_to_be_granted>
   The ID of the user created in step 1, who is going to be granted the access to the reports

   <database_name>
   The name of the database, as created when the master domain manager was installed

   [*<database_admin_user>* *<password>***]**
   The user ID and password of the database administration user. If you are running this command as the database administration user, you can omit these parameters.
3. Log on to the Dynamic Workload Console.
4. In the Portfolio, select **Manage Engines**. The Manage Engines panel is displayed.
5. Select the engine you defined or create another engine. The Engine Connection properties panel is displayed.
6. In Database Configuration for Reporting, do the following:
   a. Check **Enable Reporting** to enable the engine connection you selected to run reports.
   b. In **Database User ID and Password**, specify the database user and password that you authorized to access reports.

Configuring for an Oracle database

About this task

Actions taken on IBM Workload Scheduler engine:

For Oracle, the IT administrator, or the IBM Workload Scheduler IT administrator, or both working together, do the following:

1. Use the **TWS Oracle user** specified during the master domain manager installation or perform the following steps to create a new user:
   a. Create a database user authorized to access the database and specify a password.
   b. Launch the following script:

```
<TWA_home>/TWS/dbtools/Oracle/scripts/dbgrant.bat/.sh
  <ID_of_user_to_be_granted>
  <database_name>
  <database_admin_user> <password>
```

where the variables are as follows:

<TWA_home>
The IBM Workload Automation instance directory

<ID_of_user_to_be_granted>
The ID of the user created in step 1a on page 179, who is going to be granted the access to the reports

<database_name>
The name of the database, as created when the master domain manager was installed

<database_schema_owner> <password>
The user ID and password of the database schema owner.

2. Define a valid connection string to the database:
   a. Ensure that the following property is set in the *<TWA_home>*/WAS/TWSprofile/properties/TWSConfig.properties file to point to the Oracle JDBC URL:

      ```
      com.ibm.tws.webui.oracleJdbcURL
      ```

      For example:

      ```
      com.ibm.tws.webui.oracleJdbcURL=
                          jdbc:oracle:thin:@//9.132.235.7:1521/orcl
      ```

      The Oracle JDBC URL is also to be specified in the **PARAM_DataSourceUrl** property in the .\config\common.properties file. The common.properties file is required when setting up for command line reporting. For more information about this file, see "Setting up for command line audit reporting" on page 402 and the section about setting up for command line batch reporting in *IBM Workload Scheduler: User's Guide and Reference*.
   b. Restart the WebSphere Application Server.

Actions taken on the Dynamic Workload Console:

1. Download the JDBC drivers required by your Oracle server version.
2. Copy the JDBC drivers in a directory that is accessible by the WebSphere Application Server used by your Dynamic Workload Console.
3. Create a shared library on WebSphere Application Server specifying the path and filename of the JDBC drivers you have copied, as described in:

   WebSphere Application Server (Distributed operating systems), Version 8.0 documentation, section about Setting up the application serving environment > Administering application servers > Managing shared libraries.
4. Associate the **isc** Enterprise Application to this shared library, as described in:

   WebSphere Application Server (Distributed operating systems), Version 8.0 documentation, section about Setting up the application serving environment > Administering application servers > Managing shared libraries.
5. Log on to the Dynamic Workload Console.
6. In Dashboard Application Services Hub navigation bar, select **System Configuration** > **Manage Engines**. The Manage Engines panels opens.
7. Select the engine you defined or create another engine. The Engine Connection properties panel is displayed.

8. In Database Configuration for Reporting, do the following:
   a. Check **Enable Reporting** to enable the engine connection you selected to run reports.
   b. In **Database User ID and Password**, specify the database user and password that you authorized to access reports.

Chapter 4. Configuring user authorization (Security file)

This chapter describes how to manage the authorizations to access scheduling objects assigned to IBM Workload Scheduler users.

Getting started with security

The way IBM Workload Scheduler manages security is controlled by a configuration file named ***security file***. This file controls activities such as:

- Linking workstations.
- Accessing command-line interface programs and the Dynamic Workload Console.
- Performing operations on scheduling objects in the database or in the plan.

A template file named *TWA_home*/TWS/config/Security.conf is provided with the product. During installation, a copy of the template file is installed as *TWA_home*/TWS/Security.conf, and a compiled, operational copy is installed as *TWA_home*/TWS/Security.

This version of the file contains some predefined access definitions:

- A full access definition for the user who installed the product, *TWS_user*.
- An access definition for the system administrator (root on UNIX or Administrator on Windows.
- The following access definitions for the Dynamic Workload Console:
  - Analyst
  - Administrator
  - Configurator
  - Operator
  - Developer

As you continue to work with the product, you might want to add more users with different roles and authorization to perform specific operations on a defined set of objects.

You can update your *security file* according to the role-based security model. The role-based security model allows you to update your *security file* with the security objects (domains, roles, and access control lists) that you define in the master domain manager database. You can define your security objects by using the **Manage Workload Security** interface from Dynamic Workload Console or the **composer** command-line program. Enable the role-based security model by setting the **optman** enRoleBasedSecurityFileCreation global option to *yes*. For details about updating the security file according to the role-based security model, see Role-based security model

If you are upgrading IBM Workload Scheduler version 9.1 or earlier, you might want to continue to use the classic security model that allows you to update the security file by using **dumpsec** and **makesec** commands from the command line. To continue to use the classic security model, the enRoleBasedSecurityFileCreation global option must be set to *no* (default value). At any time, specify *yes* if you want to enable the role-based security model and replace your current security file. A

new security file is then created and updated with the security objects (domains, roles, and access control lists) that you define in the master domain manager database by using the **Manage Workload Security** interface from Dynamic Workload Console or the **composer** command-line program. For details about updating the security file according to the classic security model, see Classic security model

Changes to `enRoleBasedSecurityFileCreation` global option are effective immediately. For details about the `enRoleBasedSecurityFileCreation` global option, see Global options - detailed description.

Note: The role-based security model and the classic security model are mutually exclusive.

Role-based security model

The security objects that you define by using the **Manage Workload Security** interface from Dynamic Workload Console, or the **composer** command-line program, are:

Security roles
Each role represents a certain level of authorization and includes the set of actions that users or groups can do.

Security domains
Each domain represents the set of scheduling objects that users or groups can manage.

Access control lists
Each access control list is defined assigning roles to users or groups, on a certain security domain.

You save the definitions of your security objects in the master domain manager database. If the role-based security model is enabled for your system (see Getting started with security), whenever you need to update the security objects, your *security file* is updated consequently and converted into an encrypted format (for performance and security), replacing the previous file. The system uses this encrypted *security file* from that point onwards.

Each time a user runs IBM Workload Scheduler programs, commands, and user interfaces, the product compares the name of the user with the user definitions in the *security file* to determine if the user has permission to perform those activities, on the specified scheduling objects, in a certain security domain.

When the security file is updated on the master domain manager, the security settings on the master domain manager are automatically synchronized with the backup master domain manager.

Note: The role-based security model does not support centralized security management on fault-tolerant agents. On fault-tolerant agents the security is managed locally on each workstation.

Configuring role-based security from Dynamic Workload Console

About this task

This section explains how to create and modify the security objects by using the **Manage Workload Security** interface from Dynamic Workload Console.

To create or modify security objects, you must have permission for the **modify** action on the object type **file** with attribute **name=security**.

When working with the role-based security from Dynamic Workload Console, be aware that access to security objects is controlled by an "optimistic locking" policy. When a security object is accessed by user "A", it is not actually locked. The security object is locked only when the object update is saved by user "A", and then it is unlocked immediately afterwards. If in the meantime, the object is accessed also by user "B", he receives a warning message saying that the object has just been updated by user "A", and asking him if he wants to override the changes made by user "A", or refresh the object and make his changes to the updated object.

Managing security roles

About this task

A security role represents a certain level of authorization and includes the set of actions that users or groups can perform on a set of object types.

For the list of actions that users or groups can perform on the different objects, for each IBM Workload Scheduler task, see “Actions on security objects” on page 195.

A set of predefined security roles is available in the master domain manager database after the product has been installed:

- A full access definition for the user who installed the product, TWS_user.
- An access definition for the system administrator, root on UNIX or Administrator on Windows.
- The following access definitions for the Dynamic Workload Console:
  - Analyst
  - Administrator
  - Configurator
  - Operator
  - Developer

You can create new security roles or manage existing security roles.

Create new role:
About this task

To create a new security role from the Dynamic Workload Console, complete the following procedure:

Procedure

1. From the navigation toolbar, click **Administration**.
2. In the **Workload Environment Design**, select **Manage Workload Security**. The Manage Workload Security panel opens.

3. From the drop-down list, select the IBM Workload Scheduler engine on which you want to manage security settings.
4. In the Roles section, click **Create new role**. The Create Role panel opens.
5. Enter the name of the security role that you are creating and, optionally, the role description.
6. For each of the IBM Workload Scheduler task, assign to the security role the level of access for performing certain actions on specific object types. You can assign a predefined or a custom level of access.
7. Click **Show Details** to see the permissions associated to a predefined level of access, or to define your custom level of access. Tooltips are available to explain what a certain permission means for a particular object type.
8. Click **View** to see the mapping between the set of permissions that you are assigning and the corresponding set of permissions in the classic security model.
9. Click **Save** to save the security role definition in the database.
10. Click **Save and Exit** to save the security role definition in the database and return to the Manage Workload Security panel.

Results

The security role has now been added to the database. If the **optman** `enRoleBasedSecurityFileCreation` global option is set to *yes*, the security role is activated in your security file.

Manage roles:
About this task

From Manage Workload Security, you can also **remove**, **edit**, and **duplicate** existing roles.

Procedure

1. In the Roles section of the Manage Workload Security panel, click **Manage roles**. The list of the available security roles is displayed.
2. Select the security roles that you want to manage.
3. Select the action that you want to run on the selected roles.

Managing security domains
About this task

A security domain represents the set of objects that users or groups can manage. For example, you can define a domain that contains all objects named with a prefix 'AA'. If you want to specify different security attributes for some or all of your users, you can create additional security domains based on specific matching criteria.

You can filter objects by specifying one or more attributes for each security object type. You can include or exclude each attribute from the selection. For example, you can restrict access to a set of objects having the same name or being defined on the same workstation, or both.

For the attributes that you can specify for each security object type, see Attributes for object types.

For the values that you can specify for each object attribute, see Specifying object attribute values.

You can create new security domains or manage existing security domains.

Create new security domain:
About this task

To create a new security domain from the Dynamic Workload Console, complete the following procedure:

Procedure

1. From the navigation toolbar, click **Administration**.
2. In the **Workload Environment Design**, select **Manage Workload Security** . The Manage Workload Security panel opens.
3. From the drop-down list, select the IBM Workload Scheduler engine on which you want to manage security settings.
4. In the Security Domains section, click **Create new Security Domain**. The security domain creation panel opens.
5. Enter the name of the security domain that you are creating and, optionally, the domain description.
6. Select the type of security domain that you want to define:

   Simple
   To define a filtering rule that applies to all object types.

   Complex
   To define different filtering rules for different object types.
7. Use object filtering to select the set of security objects that users or groups can manage in the security domains that you are defining. You can use the wildcard character (*) when defining object attributes.
8. Click **View** to see the mapping between the set of security objects that you are assigning to the domain and the corresponding set of security objects in the classic security model.
9. Click **Save** to save the security domain definition in the database.
10. Click **Save and Exit** to save the security domain definition in the database and then exit.

Results

The security domain has now been added to the database. If the **optman**`enRoleBasedSecurityFileCreation` global option is set to *yes*, the security domain is activated in your security file.

Manage security domain:
About this task

From Manage Workload Security, you can also **remove**, **edit**, and **duplicate** existing security domains.

Procedure

1. In the Security Domains section of the Manage Workload Security panel, click **Manage Security Domain**. The list of the available security domains is displayed.

2. Select the security domains that you want to manage.
3. Select the action that you want to run on the selected security domains.

Managing access control list
About this task

Create an access control list by assigning security roles to users or groups, in a certain security domain .

You can:

- Give access to user or group.
- View access for user or group.
- View access for Security Domain .
- Manage accesses.

Give access to user or group:
About this task

To give access to users or groups complete the following procedure:

Procedure

1. From the navigation toolbar, click **Administration**.
2. In the **Workload Environment Design**, select **Manage Workload Security**. The Manage Workload Security panel opens.
3. From the drop-down list, select the IBM Workload Scheduler engine on which you want to manage security settings.
4. In the Access Control List section, click **Give access to user or group**. The Create Access Control List panel opens.
5. Enter the user name or the group name, the assigned roles, and the security domain.
6. Click **Save** to save the access definition in the database.
7. Click **Save and Create New** to save the access definition in the database and proceed to create a new access definition.
8. Click **Save and Exit** to save the access definition in the database and return to the Manage Workload Security panel.

Results

The access definition has now been added to the database. If the **optman** `enRoleBasedSecurityFileCreation` global option is set to *yes*, the access definition is activated in your security file.

View access for user or group:
About this task

From Manage Workload Security, you can also view the access for users or groups.

Procedure

1. In the Access Control List section of the Manage Workload Security panel, click **View access for user or group**. The input field for the user or group name is displayed.

2. Enter the user or group name and click **View**. The user or group access, with the assigned roles, to the related security domains is displayed.

View access for Security Domain:
About this task

From Manage Workload Security, you can also view the access to a certain security domain.

Procedure

1. In the Access Control section of the Manage Workload Security panel, click **View access for Security Domain**. The input field for the security domain name is displayed.
2. Enter the security domain name and click **View**. The list of users or groups, with the assigned roles, that have access to the specified security domain is displayed.

Manage accesses:
About this task

From Manage Workload Security, you can also **remove** and **edit** existing access control lists.

Procedure

1. In the Access Control List section of the Manage Workload Security panel, click **Manage Accesses**. The list of users or groups, with the assigned roles, that have access to the different security domains is displayed.
2. Select the access control list that you want to manage.
3. Select the action that you want to run on the selected access control list.

   If you select the **edit** action, you can change only the roles associated with the access control list. You cannot change the associated domain. If you want to change the domain, you must **remove** the access control list and redefine the access control list with a new domain.

Configuring role-based security with composer command-line

About this task

This section explains how to create or modify the security objects in the database, by using the **composer** command line interface.

To define security objects in the database, see:

Access control list definition

Security domain definition

Security role definition

To manage security objects in the database, see the section about managing objects with composer command-line, in the *User's Guide and Reference*.

To define or modify security objects, you must have permission for the **modify** action on the object type **file** with attribute **name=security**.

Security access control list definition

In the role-based security model, an access control list assigns security roles to users or groups, in a certain security domain. You can include multiple security access control list definitions in the same text file, along with security domain definitions and security role definitions.

Each security access control list definition has the following format and arguments:

Syntax

accesscontrollist for *security_domain_name*
user_or_group_name [*security_role*[, *security_role*]...]
[*user_or_group_name* [*security_role*[, *security_role*]...]]...
end

[**securitydomain** ...]

[**securityrole** ...]

Arguments

security_domain_name
Specifies the name of the security domain on which you are defining the access control list.

user_or_group_name [*security_role*[**,** *security_role*]
Assigns one or more security roles to a certain user or group, on the specified security domain.

Examples

The following example defines an access control list on SECDOM1 domain and an access control list on SECDOM2 domain:

```
ACCESSCONTROLLIST FOR SECDOM1
  USER1 SECROLE1, SECROLE2, SECROLE3
  USER2 SECROLE4
  USER3 SECROLE2, SECROLE4
END

ACCESSCONTROLLIST FOR SECDOM2
  USER1 SECROLE1, SECROLE2
  USER2 SECROLE3
END
```

Security domain definition

In the role-based security model, a security domain represents the set of objects that users or groups can manage. For example, you can define a domain that contains all objects named with a prefix 'AA'. If you want to specify different security attributes for some or all of your users, you can create additional security domains based on specific matching criteria. You can filter objects by specifying one or more attributes for each security object type. You can include or exclude each attribute from the selection. For example, you can restrict access to a set of objects having the same name or being defined on the same workstation, or both.

You can include multiple security domain definitions in the same text file, along with security role definitions and access control list definitions.

Each security domain definition has the following format and arguments:

Syntax

Each attribute can be included or excluded from the selection using the plus (+) and tilde (~) symbols.

securitydomain *security_domain_name*
  [**description** *"description"*]
    [**common** [[+|~]*object_attribute* [= *value* | @[, *value* | @]...]]]
    object_type [[+|~]*object_attribute* [= *value* | @[, *value* | @]...]]
    [*object_type* [[+|~]*object_attribute* [= *value* | @[, *value* | @]...]]]...
  end

[**securityrole** ...]

[**accesscontrollist** ...]

Arguments

securitydomain*security_domain_name*
: Specifies the name of the security domain. The name must start with a letter, and can contain alphanumeric characters, dashes, and underscores. It can contain up to 16 characters.

description "*description***"**
: Provides a description of the security domain. The description can contain up to 120 alphanumeric characters. The text must be enclosed within double quotes.

common [[+|~]*object_attribute* **[=** *value* | @**[,** *value* | @**]...]]**
: Provides object attributes that are common to all the security object types.

object_type **[[+|~]***object_attribute* **[=** *value* | @**[,** *value* | @**]...]]**
: For each object type, specifies the attributes that apply to that object type and the related values. Each attribute can be included or excluded from the selection using the plus (+) and tilde (~) symbols. Wildcard (@) is supported for the attribute value: *object_attribute* =@ means that all the objects matching the object attribute must be included in the domain. For the use of wildcard (@), see the examples below.

For the attributes that you can specify for each security object type, see the section about managing security with the Dynamic Workload Console, in the *Dynamic Workload Console User's Guide*.

For the values that you can specify for each object attribute, see the section about managing security with the Dynamic Workload Console, in the *Dynamic Workload Console User's Guide*.

Examples

The following example defines a security domain named SECDOM1 and a security domain named SECDOM2:

```
securitydomain SECDOM1
description "Sample Security Domain1"
job       cpu =   $THISCPU, # The workstation where the user logs on
                  $MASTER,  # The master workstation
                  $SLAVES,  # Any fault tolerant agent
                  $REMOTES  # Any standard agent
                  cogs@     # Any workstation whose name starts with "cogs"
        + name =  A@        # Any job whose name starts with "A"
```

```
        ☖ name =  A2@       # but doesn't start with A2
        + jcltype = SCRIPTNAME # Allow only SCRIPTNAME type of job definition
        + jcltype = DOCOMMAND  # Allow only DOCOMMAND type of job definition
        + logon =   $USER,  # Streamlogon is the conman/composer user
                    $OWNER, # Streamlogon is the job creator
                    $JCLOWNER, # Streamlogon is the OS owner of the file
                    $JCLGROUP  # Streamlogon is the OS group of the file
        ☖ logon =   root, twsuser  # The job cannot logon as "root" or "twsuser"
        + jcl   =   "/usr/local/bin/@"  # The jobs whose executable file that is
           present in /usr/local/bin
        ☖ jcl   =  "@rm@" # but whose JSDL definition does not contain the
           string "rm"
end

securitydomain SECDOM2
description "Sample Security Domain2"
    common      cpu=@+name=@
    userobj     cpu=@
    job         cpu=@
    schedule    cpu=@+name=AP@
    resource    cpu=@
    prompt
    file        name=@
    cpu         cpu=@
    parameter   cpu=@
    calendar
    report      name=@
    eventrule   name=@
    action      provider=@
    event       provider=@
    vartable    name=@
    wkldapp     name=@
    runcygrp    name=@
    lob         name=@
end
```

Security role definition

In the role-based security model, a security role represents a certain level of authorization and includes the set of actions that users or groups can do. You can include multiple security role definitions in the same text file, along with security domain definitions and access control list definitions.

Each security role definition has the following format and arguments:

Syntax

securityrole *security_role_name*
  [**description** *"description"*]
    object_type **access**[=*action*[,*action*]...]
    [*object_type* **access**[=*action*[,*action*]...]]...
  end

[**securitydomain** ...]

[**accesscontrollist** ...]

Arguments

securityrole*securityrolename*
: Specifies the name of the security role. The name must start with a letter, and can contain alphanumeric characters, dashes, and underscores. It can contain up to 16 characters.

description *"description"*
Provides a description of the security role. The description can contain up to 120 alphanumeric characters. The text must be enclosed within double quotes.

object_type **access[=***action***[,***action***]...]**
For each object type, specifies a list of actions that users or groups can perform on that specific object type.

Table 32 shows the different object types and how they are referenced with **composer** and with the Dynamic Workload Console:

Table 32. Security object types

Object type - composer	Object type - Dynamic Workload Console	Description
action	Actions	Actions defined in scheduling event rules
calendar	Calendars	User calendars
cpu	Workstations	Workstations, domains, and workstation classes
event	Events	Event conditions in scheduling event rules
eventrule	Event Rules	Scheduling event rule definitions
file	Files	IBM Workload Scheduler database files
job	Jobs	Scheduled jobs and job definitions
lob	IBM Application Lab	IBM Application Lab
parameter	Parameters	Local parameters
prompt	Prompts	Global prompts
report	Reports	The following reports in Dynamic Workload Console: **RUNHIST** Job Run History **RUNSTATS** Job Run Statistics **WWS** Workstation Workload Summary **WWR** Workstation Workload Runtimes **SQL** Custom SQL **ACTPROD** Actual production details (for current and archived plans) **PLAPROD** Planned production details (for trial and forecast plans)
resource	Recources	Scheduling resources
runcygrp	Run Cycle Groups	Run cycle groups
schedule	Job Streams	Job streams
userobj	User Objects	User objects

Table 32. Security object types (continued)

Object type - composer	Object type - Dynamic Workload Console	Description
vartable	Variable Tables	Variable tables
wkldappl	Workload Application	Workload application

Table 33 shows the actions that users or groups can perform on the different objects.

Table 33. Actions that users or groups can perform on the different objects

Actions that users or groups can perform on the different objects			
add	deldep	manage	shutdown
adddep	delete	modify	start
altpass	display	release	stop
altpri	fence	reply	submit
build	kill	rerun	submitdb
cancel	limit	resetfta	unlink
confirm	link	resource	unlock
console	list	run	use

For the actions that users or groups can perform on a specific object type, for each of the IBM Workload Scheduler task, see the section about managing security roles with the Dynamic Workload Console, in the *Dynamic Workload Console User's Guide*.

Examples

The following example defines security role SECROLE1 and security role SECROLE2:

```
SECURITYROLE SECROLE1
 DESCRIPTION "Sample Security Role"
 SCHEDULE  ACCESS=ADD,ADDDEP,ALTPRI,CANCEL,DELDEP,DELETE,DISPLAY,LIMIT,MODIFY,
  RELEASE
 RESOURCE   ACCESS=ADD,DELETE,DISPLAY,MODIFY,RESOURCE,USE,LIST,UNLOCK
 PROMPT        ACCESS=ADD,DELETE,DISPLAY,MODIFY,REPLY,USE,LIST,UNLOCK
 FILE        ACCESS=BUILD,DELETE,DISPLAY,MODIFY,UNLOCK
 CPU        ACCESS=LIMIT,LINK,MODIFY,SHUTDOWN,START,STOP,UNLINK,LIST,UNLOCK,RUN
 PARAMETER ACCESS=ADD,DELETE,DISPLAY,MODIFY,LIST,UNLOCK
 CALENDAR    ACCESS=ADD,DELETE,DISPLAY,MODIFY,USE,LIST,UNLOCK
 REPORT       ACCESS=DISPLAY
 EVENTRULE   ACCESS=ADD,DELETE,DISPLAY,MODIFY,LIST,UNLOCK
 ACTION       ACCESS=DISPLAY,SUBMIT,USE,LIST
 EVENT     ACCESS=USE
 VARTABLE     ACCESS=ADD,DELETE,DISPLAY,MODIFY,USE,LIST,UNLOCK
 WKLDAPPL    ACCESS=ADD,DELETE,DISPLAY,MODIFY,LIST,UNLOCK
 RUNCYGRP    ACCESS=ADD,DELETE,DISPLAY,MODIFY,USE,LIST,UNLOCK
 LOB         ACCESS=USE
END

SECURITYROLE SECROLE2
 DESCRIPTION "Sample Security Role"
 SCHEDULE  ACCESS=ADD,ADDDEP,ALTPRI,CANCEL,DELDEP,DELETE,DISPLAY,LIMIT,MODIFY,
```

```
  RELEASE
 RESOURCE   ACCESS=ADD,DELETE,DISPLAY,MODIFY,RESOURCE,USE,LIST,UNLOCK
 PROMPT        ACCESS=ADD,DELETE,DISPLAY,MODIFY,REPLY,USE,LIST,UNLOCK
END
```

Actions on security objects

The following tables show the actions that users or groups can perform on the different object types, for each IBM Workload Scheduler task. See in parenthesis the corresponding *actions* and *objects* values that you must use when defining role-based security with **composer** command line interface.

Table 34. Actions that users or groups can perform when designing and monitoring the workload

Design and Monitor Workload	
Actions that users or groups can perform	**Security object types**
List (list) Display (display) Create (add) Delete (delete) Modify (modify) Use (use) Unlock (unlock) Actions on remote workstations while modeling jobs (cpu-run) **Note:** See in parenthesis the corresponding *actions* and *objects* values that you must use when defining role-based security with **composer** command line interface.	Jobs (job) Job Streams (schedule) User Objects (userobj) Prompts (prompt) Resources (resource) Calendars (calendar) Run Cycle Groups (runcygrp) Variable Tables (vartable) Workload Application (wkldappl) Parameters (parameter)

Table 35. Actions that users or groups can perform when modifying current plan

Modify current plan
Actions that users or groups can perform on the current plan
Add job stream dependency (schedule - adddep)
Add job dependency (job - adddep)
Remove job dependency (job - deldep)
Remove job stream dependency (schedule - deldep)
Change job priority (job - altpri)
Change job stream priority (schedule - altpri)
Cancel job (job - cancel)
Cancel job stream (schedule - cancel)
Rerun job (job - rerun)
Confirm job (job - confirm)
Release job (job - release)
Release job stream (schedule - release)
Kill jobs (job - kill)
Reply to prompts (prompt - reply)
Reply to job prompts (job - reply)
Reply to job stream prompts (schedule - reply)
Alter user password (userobj - altpass)
Change jobs limit (schedule - limit)
Actions on job remote system (job - run)
Change resource quantity (resource - resource)
Note: See in parenthesis the corresponding *actions* and *objects* values that you must use when defining role-based security with **composer** command line interface.

Table 36. Actions that users or groups can perform when submitting workload

Submit Workload
Workload definitions that can be added to the current plan
Only existing job definitions (job - submitdb)
Existing jobs definitions and ad hoc jobs (job - submit)
Existing job stream definitions (schedule - submit)
Note: See in parenthesis the corresponding *actions* and *objects* values that you must use when defining role-based security with **composer** command line interface.

Table 37. Actions that users or groups can perform when managing workload environment

Manage Workload Environment
Actions that users or groups can perform on workstations, domains, and workstation classes
List workstations (cpu - list)
Display workstation details (cpu - display)
Create workstations (cpu - add)
Delete workstations (cpu - delete)
Modify workstations (cpu - modify)
Use workstations (cpu - use)
Unlock workstations (cpu - unlock)
Start a workstation (cpu - start)
Stop a workstation (cpu - stop)
Change limit (cpu - limit)
Change fence (cpu - fence)
Shutdown (cpu - shutdown)
Reset FTA (cpu - resetfta)
Link (cpu - link)
Unlink (cpu - unlink)
Use 'console' command from conman (cpu - console)
Upgrade workstation (cpu - manage)
Note: See in parenthesis the corresponding *actions* and *objects* values that you must use when defining role-based security with **composer** command line interface.

Table 38. Actions that users or groups can perform when managing event rules

Manage Event Rules
Actions that users or groups can perform on event rules
List event rules (eventrule - list)
Display event rules details (eventrule - display)
Create event rules (eventrule - add)
Delete event rules (eventrule - delete)
Modify event rules (eventrule - modify)
Use event rules (eventrule - use)
Unlock event rules (eventrule - unlock)
Display actions in the event rules (action - display)
Monitor triggered actions (action - list)
Use action types in the event rules (action - use)
Submit action (action - submit)
Use events in the event rules (event - use)
Use a File Monitor event on the workstation where the file resides. (event - display)
Note: See in parenthesis the corresponding *actions* and *objects* values that you must use when defining role-based security with **composer** command line interface.

Table 39. Administrative tasks that users or groups can perform

Administrative Tasks
Administrative tasks that users or groups can perform
View configuration (dump security and global options) (file - display)
Change configuration (makesec, optman add) (file - modify)
Delete objects definitions (file - delete)
Unlock objects definitions (file - unlock)
Allow planman deploy, prodsked and stageman (file - build)
Note: See in parenthesis the corresponding *actions* and *objects* values that you must use when defining role-based security with **composer** command line interface.

Table 40. Actions that users or groups can perform on workload reports

Workload Reports	
Actions that users or groups can perform on workload reports	
Generate workload reports (display report)	**Reports in Dynamic Workload Console** **RUNHIST** Job Run History **RUNSTATS** Job Run Statistics **WWS** Workstation Workload Summary **WWR** Workstation Workload Runtimes **SQL** Custom SQL **ACTPROD** Actual production details (for current and archived plans) **PLAPROD** Planned production details (for trial and forecast plans)
Note: See in parenthesis the corresponding *actions* and *objects* values that you must use when defining role-based security with **composer** command line interface.	

Table 41. Actions that users or groups can perform on Application Lab

Application Lab
Actions that users or groups can perform on Application Lab
Access Application Lab (use lob)
Note: See in parenthesis the corresponding *actions* and *objects* values that you must use when defining role-based security with **composer** command line interface.

Attributes for security object types

Table 42 shows the attributes that you can specify for each security object type (see in parenthesis the corresponding object type and object attribute that you must use when defining security objects with the **composer** command line interface).

Table 42. Attributes for security object types

Attribute / Security object type	Name (name)	Workstation (cpu)	Custom (custom)	JCL (jcl)	JCLtype (jcltype)	Logon (logon)	Provider (provider)	Type (type)	Host (host)	Port (port)
Actions (action)							✓	✓	✓	✓
Calendars (calendar)	✓									
Workstations (cpu)								✓		
Events (event)			✓				✓	✓		
Event rules (event)	✓									
Files (file)	✓									
Jobs (job)	✓	✓		✓	✓	✓				
Application Lab (lob)	✓									
Parameters (parameter)	✓	✓								
Prompts (prompt)	✓									
Reports (report)	✓									
Resource (resource)	✓	✓								
RunCycle groups (runcygrp)	✓									
Job streams (schedule)	✓	✓								

Table 42. Attributes for security object types (continued)

Attribute / Security object type	Name (name)	Workstation (cpu)	Custom (custom)	JCL (jcl)	JCLtype (jcltype)	Logon (logon)	Provider (provider)	Type (type)	Host (host)	Port (port)
User objects (userobj)		✓				✓				
Variable tables (vartable)	✓									
Workload applications (wkldappl)	✓									

For the values that are allowed for each object attribute, see "Specifying object attribute values."

Specifying object attribute values

The following values are allowed for each object attribute (see in parenthesis the corresponding object type and object attribute for the **composer** command line interface):

Name (name)

Specifies one or more names for the object type.

- For the **Files (file)** object type, the following values apply:

  globalopts

  Allows the user to set global options with the `optman` command. The following access types are allowed:
  - Display access for `optman ls` and `optman show`
  - Modify access for `optman chg`

  prodsked

  Allows the user to create, extend, or reset the production plan.

  security

  Allows the user to manage the security file.

  Symphony

  Allows the user to run **stageman** and **JnextPlan**.

  trialsked

  Allows the user to create trial and forecast plans or to extend trial plans.

  Note: Users who have restricted access to files should be given at least the following privilege to be able to display other object types that is, Calendars (calendar) and Workstations (cpu):

  ```
  file  name=globalopts  action=display
  ```

- For the **Variable Tables (vartable)** object type, you can use the $DEFAULT value for the **Name (name)** attribute to indicate the default variable table. This selects the table that is defined with the `isdefault` attribute.

Workstation (cpu)

Specifies one or more workstation, domain, or workstation class name. If this attribute is not specified, all defined workstations and domains can be accessed. Workstation variables can be used:

$MASTER

The IBM Workload Scheduler master domain manager.

$SLAVES

Any fault-tolerant agent.

$REMOTES

Any standard agent.

$THISCPU
The workstation on which the user is running the IBM Workload Scheduler command or program.

If you use **composer** command line to define security domains, the following syntax applies:

```
cpu=workstation[,workstation]...
```

Custom (custom)

Use this attribute to assign access rights to events defined in event plug-ins. The precise syntax of the value depends on the plug-in. For example:

- Specify different rights for different users based on SAP R/3 event names when defining event rules for SAP R/3 events.
- Define your own security attribute for your custom-made event providers.
- Specify the type of event that is to be monitored. Every event can refer to an event provider.

If you use **composer** command line to define security domains, the following syntax applies:

```
custom=value[,value]...
```

JCL (jcl)
Specifies the command or the path name of a job object's executable file. If omitted, all defined job files and commands qualify.

You can also specify a string that is contained in the task string of a JSDL definition to be used for pattern matching.

If you use **composer** command line to define security domains, the following syntax applies:

```
jcl="path" | "command" | "jsdl"
```

JCL Type (jcltype)
Specifies that the user is allowed to act on the definitions of jobs that run only scripts (if set to **scriptname**) or commands (if set to **docommand**). Use this optional attribute to restrict user authorization to actions on the definitions of jobs of one type only. Actions are granted for both scripts and commands when **JCL Type (jcltype)** is missing.

A user who is not granted authorization to work on job definitions that run either a command or a script is returned a security error message when attempting to run an action on them.

If you use **composer** command line to define security domains, the following syntax applies:

```
jcltype=[scriptname | docommand]
```

Logon (logon)

Specifies the user IDs. If omitted, all user IDs qualify.

You can use the following values for the **Logon (logon)** attribute to indicate default logon:

$USER
Streamlogon is the conman/composer user.

$OWNER
Streamlogon is the job creator.

$JCLOWNER
Streamlogon is the OS owner of the file.

$JCLGROUP
Streamlogon is the OS group of the file.

If you use **composer** command line to define security domains, the following syntax applies:

```
logon=username[,username]...
```

Provider (provider)

For **Actions (action)** object types, specifies the name of the action provider.

For **Events (event)** object types, specifies the name of the event provider.

If **Provider (provider)** is not specified, no defined objects can be accessed.

If you use **composer** command line to define security domains, the following syntax applies:

```
provider=provider_name[,provider_name]...
```

Type (type)

For **Actions (action)** object types, is the `actionType`.

For **Events (event)** object types, is the `eventType`.

For **Workstations (cpu)** object types, the permitted values are those used in **composer** or the Dynamic Workload Console when defining workstations, such as `manager`, `broker`, `fta`, `agent`, `s-agent`, `x-agent`, `rem-eng`, `pool`, and `d-pool`.

Note: The value `master`, used in **conman** is mapped against the `manager` security attributes.

If **Type (type)** is not specified, all defined objects are accessed for the specified providers (this is always the case after installation or upgrade, as the type attribute is not supplied by default).

If you use **composer** command line to define security domains, the following syntax applies:

```
type=type[,type]...
```

Host (host)
For **Actions (action)** object types, specifies the TEC or SNMP host name (used for some types of actions, such as sending TEC events, or sending SNMP). If it does not apply, this field must be empty.

If you use **composer** command line to define security domains, the following syntax applies:

```
host=host_name
```

Port (port)
For **Actions (action)** object types, specifies the TEC or SNMP port number (used for some types of actions, such as sending TEC events, or sending SNMP). If it does not apply, this field must be empty.

If you use **composer** command line to define security domains, the following syntax applies:

```
port=port_number
```

Classic security model

A template file named *TWA_home*/TWS/config/Security.conf is provided with the product. During installation, a copy of the template file is installed as *TWA_home*/TWS/Security.conf, and a compiled, operational copy is installed as *TWA_home*/TWS/Security.

This version of the file contains a full access definition for the user who installed the product, *TWS_user*, and the system administrator (root on UNIX or Administrator on Windows), who are the only users defined and allowed to connect to the user interfaces and to perform all operations on all scheduling resources.

Within the IBM Workload Scheduler network, using the security file you can make a distinction between local **root** users and the **root** user on the master domain manager by allowing local **root** users to perform operations affecting only their login workstations and providing the master domain manager **root** user the authorizations to perform operations affecting any workstation across the network.

As you continue to work with the product you might want to add more users with different roles and authorization to perform specific operations on a defined set of objects.

Do not edit the original *TWA_home*/TWS/config/Security.conf template, but follow the steps described in "Updating the security file" on page 204 to make your modifications on the operational copy of the file.

Security management overview

The way IBM Workload Scheduler manages security is controlled by a configuration file named ***security file***. This file controls activities such as:

- Linking workstations.
- Accessing command-line interface programs and the Dynamic Workload Console.
- Performing operations on scheduling objects in the database or in the plan.

In the file you specify for each user what scheduling objects the user is allowed to access, and what actions the user is allowed to perform on those objects. You can determine access by object type (for example, workstations or resources) and, within an object type, by selected attributes, such as the object's name or the workstation in the object's definition. You can use wildcards to select related sets of objects. Access rights can be granted on an "included" or an "excluded" basis, or a combination of both.

Whenever you need to change access permissions you modify the configuration file and convert it into an encrypted format (for performance and security), replacing the previous file. The system uses this encrypted *security file* from that point onwards.

Each time a user runs IBM Workload Scheduler programs, commands, and user interfaces, the product compares the name of the user with the user definitions in the *security file* to determine if the user has permission to perform those activities on the specified scheduling objects.

By default, the security on scheduling objects is managed locally on each workstation. This means that the system administrator or the *TWS_user* who installed the product on that system can decide which IBM Workload Scheduler users defined on that system can access which scheduling resources in the IBM Workload Scheduler network and what actions they can perform.

Alternatively, you can centralize control of how objects are managed on each workstation. This can be configured by setting a global option. In this scenario, you configure all user permissions in the *security file* on the master domain manager. The encrypted version of the file is distributed automatically every time you run **JnextPlan**, so that all workstations have the file locally to determine the permissions of the users on that workstation.

Updating the security file

About this task

By default, every workstation in a IBM Workload Scheduler network (domain managers, fault-tolerant agents, and standard agents) has its own security file. You can maintain that file on each workstation, or, if you enable centralized security management, you can create a security file on the master domain manager and copy it to each domain manager and agent, ensuring that all IBM Workload Scheduler users are assigned the required authorization in the file (see "Centralized security management" on page 206). Whether working on an agent workstation for an individual security file, or on the master domain manager to modify a centralized file, the steps are just the same; all that changes are the number of users you are defining - just those on the local system or all in the IBM Workload Scheduler network.

Neither the IBM Workload Scheduler processes nor the WebSphere Application Server infrastructure needs to be stopped or restarted to update the security file. You just need to close any open **conman** user interfaces before running **makesec**.

To modify the security file, perform the following steps:

1. Navigate to the *TWA_home*/TWS directory from where the **dumpsec** and **makesec** commands *must* be run.
2. Run the **dumpsec** command to decrypt the current security file into an editable configuration file. See "dumpsec" on page 205.
3. Modify the contents of the editable security configuration file using the syntax described in "Configuring the security file" on page 208.
4. Close any open **conman** user interfaces using the **exit** command.
5. Stop any connectors on systems running Windows operating systems.
6. Run the **makesec** command to encrypt the security file and apply the modifications. See "makesec" on page 205.
7. If you are using local security, the file will be immediately available on the workstation where it has been updated.

   If you are using centralized security (see "Centralized security management" on page 206) you must now do the following:

   a. If you are using a backup master domain manager, copy the file to it
   b. Distribute the centralized file manually to all fault-tolerant agents in the network (not standard, extended, or broker agents), and store it in the *TWA_home*/TWS directory
   c. Run **JnextPlan** to distribute the Symphony file that corresponds to the new Security file

See the next pages for a full description of **dumpsec** and **makesec**.

dumpsec

Writes in an editable format the information contained in the compiled and encrypted security file. The output file can be edited and then used as input for the **makesec** command which compiles and activates the modified security settings.

Authorization

You must have ***display*** access to the security file and write permission in the *TWA_home*/TWS directory from where the command *must* be run.

Syntax

dumpsec –v | –u

dumpsec *security_file* [**>** *output_file*]

Comments

If no arguments are specified, the operational security file is sent to stdout. To create an editable copy of the security file, redirect the output of the command to an output file, using the redirect symbol.

Arguments

–v Displays command version information only.

–u Displays command usage information only.

security_file
Specifies the name of the security file to dump.

[> *output_file***]**
Specifies the name of the output file, If omitted, the security file is output to the stdout.

Examples

The following command dumps the operational security file (*TWA_home*/TWS/Security) to a file named **mysec**:

```
dumpsec > mysec
```

The following command dumps a security file named **sectemp** to **stdout**:

```
dumpsec sectemp
```

makesec

Compiles security definitions and installs the security file. Changes to the security file are recognized as soon as **makesec** has completed, or, in the case of centralized security, after **JnextPlan** has distributed it.

Note: Before running the **makesec** command, stop **conman**, and, on systems running Windows operating systems, any connectors.

Authorization

You must have ***modify*** access to the security file and read permission in the *TWA_home*/TWS directory from where the command *must* be run.

Syntax

makesec –v | –u

makesec [**–verify**] *in_file*

Comments

The **makesec** command compiles the specified file and installs it as the operational security file (**../***TWA_home*/TWS/Security). If the **–verify** argument is specified, the file is checked for correct syntax, but it is not compiled and installed.

Arguments

–v Displays command version information only.

–u Displays command usage information only.

–verify
Checks the syntax of the user definitions in *in_file*. The file is not compiled and installed as the security file.

in_file Specifies the name of a file or set of files containing user definitions. Syntax checking is performed automatically when the security file is installed.

Examples

Example 1: Modifying the security file definitions - full scenario

The following example shows how to modify the security file definitions:

1. An editable copy of the operational security file is created in a file named tempsec with the **dumpsec** command:
   ```
   dumpsec > tempsec
   ```
2. The user definitions are modified with a text editor:
   ```
   edit tempsec
   ```
3. The file is then compiled and installed with the **makesec** command:
   ```
   makesec tempsec
   ```

Example 2: Compiling user definitions from multiple files

The following command compiles user definitions from the fileset userdef* and replaces the operational security file:

```
makesec userdef*
```

Centralized security management

A IBM Workload Scheduler environment where centralized security management is enabled is an environment where all workstations share the same security file information contained in the security file stored on the master domain manager and the IBM Workload Scheduler administrator on the master domain manager is

the only one who can add, modify, and delete entries in the security file valid for the entire IBM Workload Scheduler environment.

This is configured with the ***enCentSec*** global option. By default the value assigned to the ***enCentSec*** option is **no**.

To set central security management, the IBM Workload Scheduler administrator must run the following steps on the master domain manager:

1. Use the **optman** command line program, to set the value assigned to the *enCentSec* global property to **yes**. For information on how to manage the global properties using **optman**, see "Setting global options" on page 8.
2. Save the information in the security file into an editable configuration file using the **dumpsec** command.
3. Set the required authorizations for all IBM Workload Scheduler users, as described in "Configuring the security file" on page 208
4. Close any open **conman** user interfaces using the **exit** command.
5. Stop any connectors on systems running Windows operating systems.
6. Compile the security file using the **makesec** command.
7. If you are using a backup master domain manager, copy the compiled security file to it as soon as possible.
8. Distribute the compiled security file to all the workstations in the environment and store it in their *TWA_home*/TWS directories.
9. Run **JnextPlan** to update the security information distributed with the Symphony file.

   The value of the checksum of the newly compiled security file is encrypted and loaded into the Symphony file and distributed to all the workstations in the IBM Workload Scheduler network.

   On each workstation, when a link is established or when a user connects to a user interface or attempts to issue commands on the plan, either with **conman** or the Dynamic Workload Console, IBM Workload Scheduler compares the value of the checksum in the security file delivered with the Symphony file with the value of the checksum of the security file stored on the workstation. If the values are equal, the operation is allowed. If the values are different, the operation fails and a security violation message is issued.

Centralized security usage notes

In a network with centralized security management, two workstations are unable to establish a connection if one of them has ***enCentSec*** turned off in its Symphony file or if their security file information does not match.

The only exception to the security file matching criteria introduced by the centralized security management mechanism is that a workstation must always accept incoming connections from its domain manager, regardless of the result of the security file matching process.

Centralized security does not apply to IBM Workload Scheduler operations for which the Symphony file is not required. Commands that do not require the Symphony file to run use the local security file. For example, the **parms** command, used to modify or display the local parameters database, continues to work according to the local security file, even if centralized security is active and the local security file differs from the centralized security rules.

If a workstation's security file is deleted and re-created, the checksum of the new security file will not match the value in the Symphony file. In addition, a run-number mechanism associated with the creation process of the Symphony file ensures prevention from tampering with the file.

Configuring the security file

In the security file you can specify which scheduling objects a user can manage and how. You define these settings by writing user definitions. A user definition is an association between a name and a set of users, the objects they can access, and the actions they can perform on the specified objects.

When defining user authorization consider that:

- When commands are issued from the **composer** command line program, the user authorizations are checked in the security file of the master domain manager since the methods used to manage the entries in the database are invoked on the master domain manager. Therefore the user must be defined:
  - As system user on the system where the master domain manager is installed.
  - In the security file on the master domain manager with the authorizations needed to run the allowed commands on the specific objects.
- When commands are issued from the **conman** command line program, the user must be authorized to run the specific commands in the security file both on the connecting workstation and on the master domain manager where the command actually runs.

The security file is parsed one line at a time, thus any given line in the security file has been assigned a maximum length of 1024 characters. Since during the encryption process (makesec) one extra character is added to any string value in order to store its length, the number of "visible" characters could actually be more or less than 1024. As an example, consider the following line:

```
CPU=@+LOGON=test1, test2
```

The actual number of characters written into the encrypted Security file is determined according to this formula:

```
"CPU=" : 2 chars (token)
"@" : 2 chars (1 + 1 one for the length)
"LOGON=" : 2 chars (token)
"test1," : 7 chars (6 + 1 for the length) (string)
"test2" : 6 chars (5 + 1 for the length) (string)
-------------------------------------------------
total : 19 chars
```

However, if counting the actual number of visible characters, there are 23 characters including the single space between test1 and test2 and the comma separating them.

Note: The "CPU" and "LOGON" each have a real length of two characters even though they actually have three and five characters respectively. This is because certain keywords are "tokenized." This can actually help reduce the apparent character count in this case.

The configuration of the security file is described in these sections:

- “Security file syntax” on page 209
- “Specifying user attributes” on page 210
- “Specifying object types” on page 216

- "Specifying object attributes" on page 217
- "Specifying access" on page 221
- "The *TWS_user* - special security file considerations" on page 236

Security file syntax

The syntax of the security file is as follows:

Syntax

[**#** *comment*]

user *definition_name user_attributes*

begin [***** *comment*]

object_type [*object_attributes*]. **access**[**=***keyword*[**,***keyword*]...]

[*object_type* [*object_attributes*]. **access**[**=***keyword*[**,***keyword*]...]]...

end | **continue**

Arguments

[# | *] *comment*
: All text following a pound sign or an asterisk and at least one space is treated as a comment. Comments are not copied into the operational security file installed by the **makesec** command.

user *definition_name*
: Specifies the name of the user definition. The name can contain up to 36 alphanumeric characters and must start with an alphabetic character.

user_attributes
: Contains one or more attributes that identify the user or users to whom the definition applies. For details of how to define user attributes, see "Specifying user attributes" on page 210.

begin
: Begins the part containing object statements and accesses within the user definition.

object_type
: Identifies the type of object (for example: workstation, resource, or prompt) to which access is to be given for the specified user or users. All object types that the specified user or users needs to access must be explicitly defined. If they are not, no access will be given. For details of how to define object types, see "Specifying object types" on page 216.

object_attributes
: Contains one or more attributes that identify the specific objects of the defined object type to which the same access is to be given. If no object attributes are defined, access is given to all objects of the defined object type. For details of how to define object attributes, see "Specifying object attributes" on page 217.

access[=*keyword***[,***keyword***]...]**
: Describes the access to the specified objects given to the selected users. If none is specified (by specifying just the keyword "access") no access is

given to the associated objects. If **access=@** then all access rights are assigned to the specified users. For details of how to define access, see "Specifying access" on page 221.

continue
Allows a user to inherit authorization from multiple *stanzas*. Add the `Continue` keyword before the `Begin` keyword of each subsequent *stanza* to request that IBM Workload Scheduler must not stop at the first *stanza*, but must continue including also the following *stanzas* that match the user definition. The user gets the accesses for the first matching entry of each *stanza*. For an example of the use of the `Continue` keyword, see "Users logged into multiple groups [continue keyword]" on page 241.

end Terminates the user definition. The users defined in the user definition that terminates with an `end` statement do not match any subsequent user definition.

Wildcards

The following wildcard characters are permitted in user definition syntax:

? Replaces one alphanumeric character.

@ Replaces zero or more alphanumeric characters.

For information about variables supplied with the product that can be used in object attributes, refer to "Using variables in object attribute definitions" on page 221. Refer to "Sample security file" on page 237 for an example on how to use variables.

Specifying user attributes

The user attributes define *who* has the access that is going to be subsequently defined. They can identify one user, a selection of users, a group of users, a selection of groups of users, or all users. You can also exclude one or more specific users or groups from a selection. As well as being identified by logon ID and group name, users can also be described by the workstation from which they log on. And finally, you can mix selection criteria, for example selecting all users in a named group that can access from a set of workstations identified by a wildcard, but excluding a specific set of users identified by their logon IDs.

A user must be uniquely identified. If different users have the same identifier, an error is issued when **makesec** command is run. You must edit the security file by using **dumpsec** command, assign a unique identifier to users, and rerun the **makesec** command.

The general syntax: You make this selection by specifying one or more user attributes. Each user attribute is specified as follows:

user_attribute_type=*value*

user_attribute_type
Can be *cpu* (workstation), *group*, or *logon*

value Identifies an individual *cpu* (workstation), *group*, or *logon*, or, by using wildcards, can identify a set of any of these.

Including or excluding: Each attribute can be *included* or *excluded* from the selection.

Thus, for each *attribute type*, your options are one of the following:

Include all
This is the default. Thus, for example, if you want to include all *groups*, you need add no user attribute with respect to any group.

Include a selection
This can be defined in one of these ways:

- By specifically including users you want to select (individuals or one or more sets)
- By specifically excluding (from the *include all* default) all users you do *not* want to select
- By specifically including a set of users and then excluding some of those contained in the set

Which of these options you choose is determined by which is easier to specify.

Using the include or exclude symbols:

Include
Precede the user attribute expression by a plus (+) sign. All users identified by the expression will be selected, unless they are also selected by an *exclude* expression. If the first attribute in your definition is an *include*, it does not need to have a (+) sign defined, because the sign is implicit.

The default (if you specify no user attributes) is to include all users, on all workstations, in all groups, so if you want to define, for example, all users except one named user, you would just supply the *exclude* definition for the one user.

Exclude
Precede the user attribute expression by a tilde (~) sign. All users identified by the expression will *never* be selected, regardless of if they are identified by any *include* expressions.

Selection expressions: You can use the following different types of selection expression:

Basis selection expressions

Include only one attribute
user_attribute_type=*value*

For example, to include one named user logon ID, and exclude all other users:

```
logon=jsmith1
```

Exclude one attribute
~*user_attribute_type*=*value*

For example, to exclude one set of logon IDs identified by a wildcard (those that start with the letter "j"), but include all others:

```
~logon=j@
```

Include only several attributes of the same type
user_attribute_type=*value*[,*value*]...

For example, to include three specific users and exclude all others:

```
logon=jsmith1,jbrown1,jjones1
```

Exclude several attributes of the same type

~user_attribute_type=value[*,value*]...

For example, to exclude three specific users and include all others:

```
~logon=jsmith1,jbrown1,jjones1
```

Complex selection expressions

Include users identified by different selection expressions

basic_selection_expression[**+***basic_selection_expression*]...

The selection expressions can be of the same or a different attribute type:

Same attribute type

An example of the same attribute type is the following, which selects all the groups beginning with the letter "j", as well as those with the letter "z":

```
group=j@+group=z@
```

If the first selection identifies 200 users, and the second 300, the total users selected is 500.

Different attribute type

An example of selection expressions of a different attribute type is the following, which selects all the groups beginning with the letter "j", as well as all users with IDs beginning with a "6":

```
group=j@+logon=6@
```

If the first selection identifies 200 users, and the second 20, of whom 5 are also in the first group, the total users selected is 5.

Exclude users identified in one selection expressions from those identified in another

basic_selection_expression[**~***basic_selection_expression*]...

Same attribute type

The selection expressions can be of the same attribute type, provided that the second is a subset of the first. An example of the same attribute type is the following, which selects all the workstations beginning with the letter "j", but excludes those with a "z" as a second letter:

```
group=j@~group=jz@
```

If the first selection identifies 200 users, and the second 20, the total users selected is 180. Note that if the second expression had not been a subset of the first, the second expression would have been ignored.

Different attribute type

Selection expressions of a different attribute type do not have to have a subset relationship, an example being the following, which selects the group "mygroup", but excludes from the selection all users in the group with IDs beginning with a "6":

```
group=mygroup~logon=6@
```

If the first selection identifies 200 users, and the second 20, of whom 5 are also in the first group, the total users selected is 195.

Multiple includes and excludes

You can link together as many include and exclude expressions as you need to identify the precise subset of users who require the same access. The overall syntax is thus:

[**~**]*user_attribute_type***=***value*[**,***value*]...
[{**+** | **~**}*user_attribute_type***=***value*[**,***value*]...

Note: Making your *first* user attribute an *exclude* means that *all* user attributes of that type are selected *except* the indicated *value*. Thus, **~***user_attribute_type=value* equates to the following:

*user_attribute_type***=@~***same_user_attribute_type=value*

However, if you use this syntax, you cannot, and do not need to, specifically add "**+***user_attribute_type***=@**", after the negated item, so you do not define:

*~user_attribute_type=value***+***same_user_attribute_type=@*

Order of user definition: You must order user definitions from most specific to least specific. IBM Workload Scheduler scans the security file from top-down, with each user ID being tested against each definition in turn. If the user ID is satisfied by the definition, it is selected, and the matching stops.

For example:

Incorrect:

```
#First User Definition in the Security File
USER TwsUser
CPU=@+LOGON=TWS_user
Begin
job name=@ access=modify
End

#Second User Definition in the Security File
USER Twsdomain:TwsUser
CPU=@+LOGON=TWSDomain\\TWS_user
Begin
job name=@ access=display
End
```

The definitions are intended to determine the following:

1. Users on all workstations with a logon of "TWS_user" will be given "modify" access to all jobs
2. Users on all workstations with a logon of "TWSDomain\TWS_user" will be given "display" access to all jobs

However, all users with a logon of "TWS_user" will satisfy the first rule, regardless of their domain, and will be given "modify" access to all jobs. This is because defining a user without its domain is a shorthand way of defining that user ID in *any* domain; it is the equivalent of "@\TWS_User". So the second rule will never be satisfied, for any user, because the matching for the "TWS_user" stops after a successful match is made.

Correct

```
#First User Definition in the Security File
USER Twsdomain:Tws_User
CPU=@+LOGON="TWSDomain\\TWS_user"
Begin
job name=@ access=display
End

#Second User Definition in the Security File
USER Tws_User
CPU=@+LOGON=TWS_user
Begin
job name=@ access=modify
End
```

By putting the more specific definition first, both object access definitions are applied correctly.

See "Sample security file" on page 237 for a practical example.

User attribute types - detailed description: The *user_attribute_types* and their associated *values* can be any of the following:

cpu={*workstation* | **@**}
where:

workstation
Specifies the workstation on which the user is logged in. Wildcard characters are permitted. The following IBM Workload Scheduler variables can be used:

$master
Means that the user is logged in on the IBM Workload Scheduler master domain manager.

$manager
Means that the user is logged in on the IBM Workload Scheduler domain manager.

$thiscpu
Means that the user is logged in on the IBM Workload Scheduler workstation on which the security check is running.

@ Specifies that the user is accessing IBM Workload Scheduler with the Dynamic Workload Console, or is logged in on any IBM Workload Scheduler workstation.

group=*groupname*
Specifies the name of the group of which the user is a member. Available for both UNIX and Windows users. Wildcard characters are permitted.

logon={*user name* | **@**}
where:

user name

Specifies the user ID with which the user is logged in on a IBM Workload Scheduler workstation. Wildcard characters are permitted. The **cpu=** attribute must be set to a specific workstation name (no wildcards) or @.

The `user name` value can have one of the following formats:

user name

The Windows user. For example if you use the `user1` value in the logon field, in the `Security` file you have the following line:

```
.............
logon=user1
.............
```

domain\user name

The user belongs to a Windows domain. Insert the escape character '\' before the '\' character in the `domain\user name` value. For example if you use the `MYDOMAIN\user1` value in the logon field, in the `Security` file you have the following line:

```
.............
logon=MYDOMAIN\\user1
.............
```

user name@internet_domain

The user belongs to an internet domain. The user name is in User Principal Name (UPN) format. UPN format is the name of a system user in an email address format. The user name is followed by the "at sign" followed by the name of the Internet domain with which the user is associated.Insert the escape character '\' before the '@' character in the `user name@internet_domain` value. For example if you use the `administrator@bvt.com` value in the logon field, in the `Security` file you have the following line:

```
.............
logon=administrator\@bvt_env.com
.............
```

For more information about the use of the wildcard with the `domain\user name` and `user name@internet_domain` format in the `Security` file, see "Sample security file" on page 237.

Note:

1. If the WebSphere Application Server security configuration option **useDomainQualifiedUserNames** is set to *true*, each user ID defined in the security file must have the format `domain\username` to use the product from one of the following:
   - **composer**
   - **Dynamic Workload Console**
   - **logman**
   - **optman**
   - **planman**

   For more information on WebSphere Application Server security configuration, see "Changing the security settings" on page 443.
2. If the user is defined on a Windows 2003 system, or when upgrading the Windows operating system from an older version to one of those mentioned above, make sure you add the **Impersonate a client after authentication** right to the user settings.

@ Specifies any user logged in with any name or being a member of any IBM administrators group.

Specifying object types

Specify one or more object types that the user or users in the associated user definition is authorized to access. If you specify the object type but no attributes, the authorized actions defined for the user with the **access** keyword apply to all the objects of that type defined in the IBM Workload Scheduler domain. If an object type from the following list is omitted for a user or users, no objects of that type can be accessed.

The object types are the following:

action Actions defined in scheduling event rules

calendars
User calendars

cpu Workstations, domains and workstation classes

event Event conditions in scheduling event rules

eventrule
Scheduling event rule definitions

file IBM Workload Scheduler database file

job Scheduled jobs and job definitions

lob IBM Application Lab

For more information, see the section about granting IBM Application Lab authorization to users in the security file in *Application Lab User's Guide*.

parameter
Local parameters. See note below.

prompt
Global prompts

report The reports on the Dynamic Workload Console that have the following *names*:

RUNHIST
Job Run History

RUNSTATS
Job Run Statistics

WWS Workstation Workload Summary

WWR Workstation Workload Runtimes

SQL Custom SQL

ACTPROD
Actual production details (for current and archived plans)

PLAPROD
Planned production details (for trial and forecast plans)

Permission to use these reports is granted by default to the *TWS_user* on fresh installations.

resource
Scheduling resources

/ **runcygrp**
/ Run cycle groups

/ **schedule**
/ Job streams

/ **userobj**
/ User objects

/ **vartable**
/ Variable tables. This includes authorization to the variable definitions in
/ the tables. See the note below.

/ **wkldappl**
/ Workload applications

Note: Starting from version 8.5, the **parameter** object type is reserved for parameters created and managed in a local parameter database with the `parms` utility command, while authorization to act on global variables is managed using the **vartable** object type. For this reason, when the security file is migrated from previous versions to 8.5, a `vartable` security definition for the default variable table is added to match each `parameter` definition found, as part of the upgrade process documented in the *IBM Workload Scheduler: Planning and Installation Guide*.

Specifying object attributes

Specify one or more attributes that identify a set of objects that the user of the user definition is authorized to access. If you specify the object type but no object sets, the authorized actions defined for the user with the **access** keyword apply to all the objects of that type defined in the IBM Workload Scheduler domain.

The general syntax: Each object attribute is specified as follows:

object_attribute=*value*

object_attribute
Object attributes differ according to the object. All objects can be selected by *name*, but some, *jobs*, for example, can be selected by the *workstation* on which they run. See "Object attribute" for full details of which attributes are available for each object type.

value Identifies an individual object, or, by using wildcards, a set of objects. See "Specifying object attributes" for full details of which attributes are available for each object type.

Object attribute: "Specifying object attributes" lists object attributes that are used to identify a specific set of object within all objects of the same type. For example, access can be restricted to a set of resource objects having the same name or being defined on the same workstation, or both.

Table 43. Object attribute types for each object type

Attribute / Object	name	cpu	custom	jcl	jcltype	logon	provider	type	host	port
action							✔	✔	✔	✔
calendar	✔									
cpu (workstation)		✔						✔		
event			✔				✔	✔		
eventrule	✔									

Table 43. Object attribute types for each object type (continued)

Attribute / Object	name	cpu	custom	jcl	jcltype	logon	provider	type	host	port
file	✓									
job	✓	✓		✓	✓	✓				
lob	✓									
parameter	✓	✓								
prompt	✓									
report	✓									
resource	✓	✓								
runcygrp	✓									
schedule (job stream)	✓	✓								
userobj		✓				✓				
vartable	✓									
wkldappl	✓									

Note: Granting access to a workstation class or a domain means to give access just to the object itself, and grant no access to the workstations in the object.

Including or excluding: Each attribute can be *included* or *excluded* from the selection using the plus (+) and tilde (~) symbols, in the same way as for the user attributes.

Selection expressions: The detailed syntax and use of the selection expressions for objects is the same as that used to select users:

[**~**]*object_attribute***=***value*[**,***value*]...[{**+** | **~**}*object_attribute***=***value*[**,***value*]...

Order of object definition: You must order object definitions from most specific to least specific, in the same way as for user attributes. For example,

Incorrect

```
job name=@ access=display
job name=ar@ access=@
```

In this case, a job with the name beginning with "ar" would satisfy the first definition, and so would be given the display access, not all access.

Correct

```
job name=ar@ access=@
job name=@ access=display
```

Ensure that you order object definitions from most specific to least specific also when you use the `Continue` keyword.

The `Continue` keyword allows a user to inherit authorization from multiple *stanzas*. The user receives accesses as defined in the first matching entry of each *stanza* that matches the user definition. For an example of a security file with the `Continue` keyword, see "Users logged into multiple groups [continue keyword]" on page 241

Specifying object attribute values: The following describes the values allowed for each object attribute type:

name=*name*[**,***name*]**...**

Specifies one or more names for the object type. Wildcard characters are permitted. Multiple names must be separated by commas.

- The following values apply to the **file** object type:

  globalopts
  Allows the user to set global options with the `optman` command. Gives the following access types:
  - Display access for `optman ls` and `optman show`
  - Modify access for `optman chg`

  prodsked
  Allows the user to create, extend, or reset the production plan.

  security
  Allows the user to manage the security file.

  Symphony
  Allows the user to run **stageman** and **JnextPlan**.

  trialsked
  Allows the user to create trial and forecast plans or to extend trial plans.

  Note: Users who have restricted access to files should be given at least the following privilege to be able to display other objects (ie. calendars and cpus):

  ```
  file  name=globalopts  access=display
  ```

- For the **event** object type use one or more of the event type names listed in the *TWSObjectsMonitor events* table or the *FileMonitor events* table in the *IBM Workload Scheduler: User's Guide and Reference*.
- For the **action** object type use one or more of the action type names listed in the table *Action types by action provider* in the *IBM Workload Scheduler: User's Guide and Reference*.
- For the **vartable** object type, you can use the $DEFAULT value for the **name** attribute to indicate the default variable table. This selects the table defined with the `isdefault` attribute.

cpu=*workstation*[**,***workstation*]**...**

Specifies one or more workstation, domain, or workstation class names. Wildcard characters are permitted. Multiple names must be separated by commas. If this attribute is not specified, all defined workstations and domains can be accessed. Workstation variables can be used - see "Using variables in object attribute definitions" on page 221.

custom=value[,*value*]**...**

Use this attribute to assign access rights to events defined in event plug-ins. The precise syntax of the value will depend on the plug-in. For example:

- Specify different rights for different users based on SAP R/3 event names when defining event rules for SAP R/3 events.
- Define your own security attribute for your custom-made event providers.
- Specify the type of event that is to be monitored. Every event can be referred to an event provider.

jcl="*path***" | "***command***" | "***jsdl***"**

Specifies the command or the path name of a job object's executable file.

The command or path must be enclosed in double quotation marks (" "). Wildcard characters are permitted. If omitted, all defined job files and commands qualify.

You can also specify a string contained in the task string of a JSDL definition to be used for pattern matching. Ensure that the string begins and ends with the @ wildcard character and that it is entirely enclosed in double quotation marks as follows: "@<*my_string*>@".

jcltype=[scriptname | docommand]
Specifies that the user is allowed to act on the definitions of jobs that run only scripts (if set to **scriptname**) or commands (if set to **docommand**). Use this optional attribute to restrict user authorization to actions on the definitions of jobs of one type or the other only. Actions are granted for both scripts and commands when **jcltype** is missing.

A user who is not granted authorization to work on job definitions that run either a command or a script is returned a security error message when attempting to run an action on them.

logon=*username***[,...]**

Specifies the user IDs. Wildcard characters are permitted. Multiple names must be separated by commas. If omitted, all user IDs qualify.

The user ID can be a Windows domain user or an internet domain user and must be defined in one of the following formats:

domain\user name
The user belongs to a Windows domain. Insert the escape character '\' before the '\' character in the `domain\user name` value. For example if you use the `MYDOMAIN\user1` value in the logon field, in the `Security` file you have the following line:

```
.............
logon=MYDOMAIN\\user1
.............
```

user name@internet_domain

The user belongs to an internet domain. The user name is in User Principal Name (UPN) format. UPN format is the name of a system user in an email address format. The user name is followed by the "at sign" followed by the name of the Internet domain with which the user is associated.Insert the escape character '\' before the '@' character in the `user name@internet_domain` value. For example if you use the `administrator@bvt.com` value in the logon field, in the `Security` file you have the following line:

```
.............
logon=administrator\@bvt_env.com
.............
```

provider=*provider_name***[,...]**

For **action** object types, specifies the name of the action provider.

For **event** object types, specifies the name of the event provider.

Wildcard characters are permitted. Multiple names must be separated by commas. If `provider` is not specified, no defined objects can be accessed.

type=*type***[,...]**

For **action** object types, is the `actionType`.

For **event** object types, is the `eventType`.

For **cpu** object types, the permitted values are those used in **composer** or the Dynamic Workload Console when defining workstations, such as `manager`, `broker`, `fta`, `agent`, `s-agent`, `x-agent`, `rem-eng`, `pool`, and `d-pool`.

Note: The value `master`, used in **conman** is mapped against the `manager` security attributes.

Wildcard characters are permitted. Multiple names must be separated by commas. If **type** is not specified, all defined objects are accessed for the specified providers (this is always the case after installation or upgrade, as the type attribute is not supplied by default).

host=*host_name*
For **action** object types, specifies the TEC or SNMP host name (used for some types of actions, such as sending TEC events, or sending SNMP). If it does not apply, this field must be empty.

port=*port_number*
For **action** object types, specifies the TEC or SNMP port number (used for some types of actions, such as sending TEC events, or sending SNMP). If it does not apply, this field must be empty.

Using variables in object attribute definitions: The following variables supplied with the product can be used in object attributes:

Workstation identifiers

$master
The IBM Workload Scheduler master domain manager.

$manager
The IBM Workload Scheduler domain manager.

$thiscpu
The workstation on which the user is running the IBM Workload Scheduler command or program.

Variable table identifiers

$default
The name of the current default variable table.

Specifying access

About this task

Specify the type of access the selected users are allowed to have to the specified objects as follows:

access[=*keyword*[,*keyword*]...]

- To specify that no actions are permitted, use **access=**
- To specify that all actions are permitted, use **access=@**
- To specify any other access, consult the access tables, by object type, below.

How the access tables are organized:
The access tables for object types are as follows:

"Object type - vartable" on page 235
This gives the access rights for variable tables, which are not included in the common table.

"Object type - workload application" on page 235
This gives the access rights for workload applications, which are not included in the common table.

Object types - calendar, cpu, eventrule, job, prompt, resource, run cycle group, schedule, userobj, vartable - using in composer:
The following table gives the access keywords required to use composer to work with objects of the following types:

- calendar
- cpu
- eventrule
- job
- prompt
- resource
- run cycle group
- schedule
- userobj
- vartable

Note:

- Starting from version 8.5, the `parameter` keyword is reserved for parameters created and managed in a local parameter database with the `parms` utility command.

  For more information about `parms`, see the related section in the *User's Guide and Reference*.

- If you plan to upgrade your environment from a previous version of IBM Workload Scheduler and use event-driven workload automation, you need to manually add the `display` access keyword to all workstations on which you plan to define File Monitor events.

  For more information about event-driven workload automation, see the related section in the *User's Guide and Reference*.

Table 44. Access keywords for composer actions

Activity			Access keywords required
Composer	add	Add new object definitions in the database from a file of object definitions. Unlock access is needed to use the `;unlock` attribute. For variable tables, to *add* individual variable entries within a table, the table must have *modify* access.	add, modify, unlock
	add event rule	Add an event rule of type File Monitor.	display
	create	Create a text file of object definitions in the database. Modify access is need to use the `;lock` attribute. For variable tables, create individual variable entries within the table.	display, modify
	delete	Delete object definitions from the database. For variable tables, to *delete* individual variable entries within a table, the table must have *modify* access.	delete
	display	Display object definitions in the database.	display
	extract	Extract a text file of object definitions from the database.	display
	list	List object definitions in the database.	If the **enListSecChk** global option is set to `yes` on the master domain manager then, either list, or list and display are required.
	lock	Lock object definitions in the database.	modify
	modify	Modify object definitions in the database. Definitions are extracted into a file. After you have edited the file the definitions are used to replace the existing ones. For variable tables, to *modify* individual variable entries within a table, the table must have *modify* access.	add, modify
	new	Create object definitions in the database from a template.	add, modify
	print	Print object definitions in the database.	display
	rename	Rename object definitions in the database. You need add access to the new object and delete and display access to the old object.	add, delete, display
	replace	Replace object definitions in the database. Unlock access is needed to use the `;unlock` attribute.	add, modify, unlock
	unlock	Unlock object definitions in the database. For variable tables, unlocking a table unlocks all the variables contained therein. Unlocking a variable unlocks the entire table where it is defined.	unlock

Table 44. Access keywords for composer actions (continued)

Activity			Access keywords required
Dynamic Workload Console	Add event rule	Add an event rule of type File Monitor.	display
	Create object in database	Add new object definitions in the database.	add
	Delete object in database	Delete object definitions from the database. Unlock access is needed to use the `;unlock` option.	delete
	Display object in database	Display object definitions in the database.	display
	List object in database	List object definitions in the database.	display
	Modify object in database	Modify object definitions in the database. Unlock access is needed to use the `;unlock` option.	modify
	Unlock object in database	Unlock object definitions in the database locked by another user.	unlock
	Perform operations for job types with advanced options, both those supplied with the product and the additional types implemented through the custom plug-ins. You can define and perform operations on job types with advanced options with the Workload Designer.	Perform operations for job types with advanced options in the database.	run
Using the workload service assurance feature	All activities	For any user to perform any workload service assurance activities, the *TWS_user* must have the following access keywords for all *cpu*, *job*, and *schedule* objects:	display, modify, list

Example: To allow a user to use the composer list, display, and modify actions on event rules, specify:

```
eventrule          access=add,display,modify
```

Object type - action:
The following table gives the access keywords required for actions:

Table 45. Actions - access keywords

Activity		Access keywords required
Dynamic Workload Console	Display action instances	display
	List action instances.	list

Table 45. Actions - access keywords (continued)

Activity		Access keywords required
Dynamic Workload Console **conman**	Use these specific action types in event rule definitions. • For actions with provider `TWSAction` and types `sbj`, `sbd`, or `sbs`, you must set this keyword in combination with the `submit` access keyword for the specific jobs and job streams specified in the action. • For actions with provider `TWSAction` and type `reply`, you must set this keyword in combination with the `reply` access keyword set for the specific prompts specified in the action. The *TWS_user* of the workstation running the event processing server must have these `submit` and `reply` authorizations, otherwise the event processing server will not be able to run this type of actions.	use

Example: To allow a user to use the Dynamic Workload Console to list action instances, specify:

```
action          access=list
```

Object type - calendar:
The following table gives the additional access keywords required to work with calendars, other than those described in Table 44 on page 224:

Table 46. Calendar - additional access keywords

Activity		Access keywords required
Composer Dynamic Workload Console	Use calendars in: • job streams • run cycles • run cycle groups	use

Example 1: To allow a user to only use calendars when working with job streams in any of the interfaces, specify:

```
calendar          access=use
```

Example 2: To allow a user to display, list, and print calendars, and use them when working with job streams in any of the interfaces, specify:

```
calendar          access=display,use,list
```

Object type - cpu:
The following table gives the additional access keywords required to work with cpus (includes workstations, domains, and workstation classes), other than those described in Table 44 on page 224:

Table 47. Cpus - additional access keywords

Activity			Access keywords required
Conman Dynamic Workload Console	console	View and send messages to the IBM Workload Scheduler **conman** console.	console
	deployconf	Force update the monitoring configuration file for the event monitoring engine.	start
	fence	Alter workstation job fences in the production plan.	fence
	limit cpu	Alter workstation job limits in the production plan.	limit
	link	Open workstation links.	link
	resetfta	Generates an updated Sinfonia file and sends it to a fault-tolerant agent on which the Symphony file has corrupted.	resetfta
	showcpus	Display workstations, domains and links in the plan.	list
	shutdown	Shut down IBM Workload Scheduler processing.	shutdown
	start	Start IBM Workload Scheduler processing.	start
	startappserver	Start the application server.	start
	starteventprocessor	Start the event processor server.	start
	startmon	Start the event monitoring engine.	start
	stop	Stop IBM Workload Scheduler processing.	stop
	stop;progressive	Stop IBM Workload Scheduler processing progressively.	stop
	stopappserver	Stop the application server.	stop
	stopeventprocessor	Stop the event processor server.	stop
	stopmon	Stop the event monitoring engine.	stop
	switcheventprocessor	Switch the event processor server from the master domain manager to the backup master domain manager or vice versa.	start, stop
	switchmgr	Switch the domain manager functionality to a workstation.	start, stop
	unlink	Close workstation links.	unlink
	upgrade	Install a fix pack or upgrade to a later version fault-tolerant agents and dynamic agents.	manage
Startup	Start IBM Workload Scheduler processing.		start
Using the workload service assurance feature	All activities	For any user to perform any workload service assurance activities, the *TWS_user* must have the following access keywords:	display, modify, list
Composer Dynamic Workload Console	Use a File Monitor event on the workstation where the file resides.		display

Note: If you plan to upgrade your environment from a previous version of IBM Workload Scheduler and use event-driven workload automation, you need to manually add the `display` access keyword to all workstations on which you plan to define File Monitor events.

For more information about event-driven workload automation, see the related section in the *User's Guide and Reference*.

Example: To allow a user to display, list, and print workstation, workstation class, and domain definitions, and link and unlink workstations, specify:

```
cpu          access=display,link,unlink
```

Object type - event:
The following table gives the access keywords required to work with events:

Table 48. Events - access keywords

Activity		Access keywords required
Composer Dynamic Workload Console	Use an event in an event rule definition.	use

Note: If you plan to upgrade your environment from a previous version of IBM Workload Scheduler and use event-driven workload automation, you need to manually add the `display` access keyword to all workstations on which you plan to define File Monitor events.

For more information about event-driven workload automation, see the related section in the *User's Guide and Reference*.

Example: To allow a user to use an event in an event rule definition, specify:

```
event          access=use
```

Object type - file:
The following table gives the access keywords required to work with files (valid only for the command line).

You must specify the file names to which the type of access applies.

Table 49. Files - access keywords

Activity			Access keywords required
dumpsec	Create a text file of the settings contained in the compiled security file.		display
JnextPlan	Generate the production plan.		build
makesec	Compile the security file from a text file of the settings.		modify
optman	ls	List all global options.	display
	show	Show the details of a global option.	display
	change	Change the details of a global option.	modify
planman	deploy	Manually deploy event rules.	build
prodsked	Work with the production plan.		build
stageman	Carry forward incompleted job streams, archive the old production plan, and install the new production plan.		build

Example 1: To allow a user to manage the globalopts file, specify:

```
file  name=globalopts    access=display,modify
```

Example 2: To allow a user to run **JnextPlan**, specify:

```
file          access=build
```

Note: The user will also be able to run **planman deploy**, **prodsked**, and **stageman**.

Object type - job:
The following table gives the additional access keywords required to work with jobs, other than those described in Table 44 on page 224:

Table 50. Jobs - additional access keywords

Activity			**Access keywords required**
Composer Dynamic Workload Console	Use jobs in job streams. Also, if a job is used as a recovery job in a job definition, the user must have "use" access to the definition of the job identified as the recovery job.		use
Conman Dynamic Workload Console	adddep	Add dependencies to jobs in the production plan. Not valid for workstations in end-to-end environment.	adddep
	altpri	Alter the priority of jobs in the production plan. Not valid for workstations in end-to-end environment.	altpri
	cancel job	Cancel jobs in the production plan. Not valid for workstations in end-to-end environment.	cancel
	confirm	Confirm completion of jobs in the production plan. Not valid for workstations in end-to-end environment.	confirm
	deldep job	Delete dependencies from jobs in the production plan. Not valid for workstations in end-to-end environment.	deldep
	display	Display jobs in the plan.	display
	Hold	Hold a job to prevent it from running	adddep
	kill	Kill running jobs.	kill
	release job	Release jobs from dependencies in the production plan. Not valid for workstations in end-to-end environment.	release
	reply	Reply to job prompts in the production plan.	reply
	rerun	Rerun jobs in the production plan. Not valid for workstations in end-to-end environment. To use the **from** argument, you must have ***submitdb*** access to the job.	rerun
	showjobs	Display information about jobs in the production plan.	list

Table 50. Jobs - additional access keywords (continued)

Activity			Access keywords required
Conman Dynamic Workload Console	submit docommand	Submit commands as jobs or recovery jobs into the production plan. If the **submit** also identifies a second job with the "ALIAS" or "RECOVERYJOB" arguments, the user must have "submit" access to that other job, as well Not valid for workstations in end-to-end environment.	submit
	submit file	Submit files as jobs or recovery jobs into the production plan. If the **submit** also identifies a second job with the "ALIAS" or "RECOVERYJOB" arguments, the user must have "submit" access to that other job, as well Not valid for workstations in end-to-end environment.	submit
	submit job	Submit jobs or recovery jobs into the production plan. If the **submit** also identifies a second job with the "ALIAS" or "RECOVERYJOB" arguments, the user must have "submit" access to that other job, as well Not valid for workstations in end-to-end environment.	submit
		Restricts the submission action to jobs defined in the database. With this authorization level a user cannot submit ad hoc jobs. Use this keyword to allow a user to submit only jobs defined in the database. Use the **submit** keyword to allow a user to submit both defined and ad hoc jobs. Users granted only **submitdb** rights: • Cannot run **submit docommand** and **submit file** successfully • Are displayed tasks related to ad hoc job submission on the graphical user interfaces, but if they run them, are returned error messages for lacking the **submit** access right.	submitdb
	submit sched	Submit job streams into the production plan. Not valid for workstations in end-to-end environment.	submit
	Hold	Hold a job to prevent it from running	adddep
Dynamic Workload Console	For critical jobs on which you run any of the following actions: • Display hot list • Display critical path • Display incompleted predecessors • Display completed predecessors	The predecessors are listed regardless of the fact that this authorization might not be extended to them. However, if you want to run any further action on any of the listed predecessors, this will require that you have the proper authorization.	list
Using the workload service assurance feature	All activities	For any user to perform any workload service assurance activities, the *TWS_user* must have the following access keywords:	display, modify, list

Example 1: To allow a user to manage only job dependencies, specify:

```
job          access=adddep,deldep
```

Example 2: To allow a user to only manage critical jobs, specify:

```
job          access=list,altpri
```

Example 3: User `administrator` is granted **add** and **modify** rights for all job definitions, and is therefore permitted to create and modify job definitions that run scripts or commands as needed, with no restriction:

```
USER TWSADMIN
CPU=@+LOGON=administrator
BEGIN
JOB CPU=@ ACCESS=ADD,MODIFY,DISPLAY,...
[...]
END
```

User `sconnor` is granted the same rights for jobs that match the condition **jcltype=scriptname**, which means that he can create or modify only job definitions that run scripts and cannot change any of them into a job that runs a command:

```
USER RESTRICTED
CPU=@+LOGON=sconnor
BEGIN
JOB CPU=@+JCLTYPE=SCRIPTNAME ACCESS=ADD,MODIFY,DISPLAY,...
[...]
END
```

Example 4: User `administrator` is granted **submit** permission for all jobs, and is therefore permitted to submit jobs defined in the database and ad hoc, with no restriction:

```
USER TWSADMIN
CPU=@+LOGON=administrator
BEGIN
JOB CPU=@ ACCESS=ADD,ADDDEP,...,RERUN,SUBMIT,USE,LIST,UNLOCK
[...]
END
```

User `jsmith` is granted **submitdb** permission for all jobs, allowing her to submit all jobs defined in the database, but she is not permitted to run ad hoc job submissions:

```
USER RESTRICTED
CPU=@+LOGON=jsmith
BEGIN
JOB CPU=@ ACCESS=ADD,ADDDEP,...,RERUN,SUBMITDB,USE,LIST,UNLOCK
[...]
END
```

Object type - lob:
The following table gives the additional access keywords required to work with IBM Application Lab, other than those described in Table 44 on page 224:

Table 51. Lob - additional access keywords

Activity	Access keywords required
Access IBM Application Lab.	use

For more information, see the section about granting Application Lab authorization to users in the security file in *Application Lab User's Guide*.

Example 1: To allow a user to access only the objects identified by the <environment_id> prefix from the Application Lab, specify:

```
lob name=<environment_id> access=use
```

Object type - parameter:
The following table gives the access keywords required to work with parameters:

Note: Starting from version 8.5, the `parameter` keyword is reserved for parameters created and managed in a local parameter database with the `parms` utility command. See the *IBM Workload Scheduler: User's Guide and Reference* for details on `parms`.

Table 52. Parameters - additional access keywords

Activity		Access keywords required
parms	Manage local parameter definitions.	display

Example: To allow a user to perform all activities on parameters, specify:

```
parameter          access=@
```

Object type - prompt:
The following table gives the additional access keywords required to work with prompts, other than those described in Table 44 on page 224:

Table 53. Prompts - additional access keywords

Activity			Access keywords required
Composer Dynamic Workload Console	Use prompts when defining or submitting jobs and job streams		use
Conman Dynamic Workload Console	adddep	Use prompts when adding dependencies to jobs in the production plan. Not valid for workstations in end-to-end environment.	use
	recall	Display prompts waiting for a response.	display
	reply	Reply to a job or Job Scheduler prompt.	reply
	showprompts	Display information about prompts.	list
	submit docommand	Use prompts when submitting commands as jobs into the production plan. Not valid for workstations in end-to-end environment.	use
	submit file	Use prompts when submitting files as jobs into the production plan. Not valid for workstations in end-to-end environment.	use
	submit job	Use prompts when submitting jobs into the production plan. Not valid for workstations in end-to-end environment.	use
	submit sched	Use prompts when submitting job streams into the production plan. Not valid for workstations in end-to-end environment.	use

Example: To allow a user to perform all activities on prompts except reply to them, specify:

```
prompt          access=use,display,list
```

Object type - report:
The following table gives the access keywords required to work with reports.

Table 54. Files- access keywords

Activity		Access keywords required
Dynamic Workload Console	Display reports on Dynamic Workload Console.	display

Example: To allow a user to display reports on the Dynamic Workload Console, specify:

```
report          access=display
```

Object type - resource:
The following table gives the additional access keywords required to work with resources, other than those described in Table 44 on page 224:

Table 55. Resources - additional access keywords

Activity			Access keywords required
Composer Dynamic Workload Console	Use resources when defining or submitting jobs and job streams		use
Conman Dynamic Workload Console	adddep	Use resources when adding dependencies to jobs in the production plan. Not valid for workstations in end-to-end environment.	use
	resource	Change the number of units of a resource on a workstation.	resource
	showresources	Display information about resources.	list
	submit docommand	Use resources when submitting commands as jobs into the production plan. Not valid for workstations in end-to-end environment.	use
	submit file	Use resources when submitting files as jobs into the production plan. Not valid for workstations in end-to-end environment.	use
	submit job	Use resources when submitting jobs into the production plan. Not valid for workstations in end-to-end environment.	use
	submit sched	Use resources when submitting job streams into the production plan. Not valid for workstations in end-to-end environment.	use

Example: To allow a user to display information about resources and change the units of a resource on a workstation, but not to use them in any other scheduling objects or actions, specify:

```
resource        access=list,resource
```

Object type - run cycle group:

The following table gives the access keywords required to work with run cycle groups:

Table 56. Run cycle groups- access keywords

Activity		Access keywords required
Composer Dynamic Workload Console	Use run cycle groups in job streams.	use

Example: To allow a user to create and delete a run cycle group, specify:

```
runcygrp          access=add,delete
```

Object type - schedule:
The following table gives the additional access keywords required to work with job streams, other than those described in Table 44 on page 224:

Table 57. Job streams - additional access keywords

Activity			Access keywords required
Conman Dynamic Workload Console	adddep	Add dependencies to job streams in the production plan. Not valid for workstations in end-to-end environment.	adddep
	altpri	Alter the priority of job streams in the production plan. Not valid for workstations in end-to-end environment.	altpri
	cancel sched	Cancel job streams in the production plan. Not valid for workstations in end-to-end environment.	cancel
	deldep sched	Delete dependencies from job streams in the production plan. Not valid for workstations in end-to-end environment.	deldep
	display	Display job streams in the plan. .	display
	limit sched	Modify the limit for jobs concurrently running within a Job Scheduler.	limit
	release sched	Release job streams from dependencies in the production plan. Not valid for workstations in end-to-end environment.	release
	reply	Reply to job stream prompts in the production plan.	reply
	showschedules	Display information about job streams in the production plan.	list
	submit sched	Submit job streams into the production plan. If the **submit** also identifies a second job stream with the "ALIAS" argument, the user must have "submit" access to that other job stream, as well Not valid for workstations in end-to-end environment.	submit
Using the workload service assurance feature	All activities	For any user to perform any workload service assurance activities, the *TWS_user* must have the following access keywords:	display, modify, list

Example: To allow a user to perform all actions on a job stream except submit it and release it, specify:

```
schedule          access=adddep,altpri,cancel,deldep,display,limit,reply,list
```

Object type - userobj:
The following table gives the additional access keywords required to work with users, other than those described in Table 44 on page 224:

Table 58. Users - additional access keywords

Activity			Access keywords required
Composer Dynamic Workload Console	Modeling of job types with advanced options	When defining job types with advanced options allows the modeler to specify in the credentials section of the job that the `user name` and `password` values required to submit the job are resolved at run time with values extracted from the database and defined with the User definition composer commands (`username` and `password`) or Dynamic Workload Console panel. Note that on dynamic agents User definitions can be used regardless of the operating system.	use
Conman Dynamic Workload Console	altpass	Alter user passwords in the plan.	altpass

Example: The following access definition allows a user to:

- List and modify user information, including passwords in the database (`display`, `modify`, and `altpass`).
- When defining job types with advanced options on dynamic agents, to specify in the credentials section of the job that the `user name` and `password` values required to submit the job are resolved at run time with values extracted from the database and defined with the User definition (`use`).

```
userobj          access=display,modify,altpass,use,list
```

Object type - vartable:
The following table gives the access keywords for using variable tables and the variables they contain (this includes the global variables)

Table 59. Variable tables - access keywords

Activity		Access keywords required
Composer Dynamic Workload Console	Use variable tables in run cycles, run cycle groups, job streams, and workstations	use

Example: To allow a user only to use variable tables when defining other scheduling objects, specify:

```
vartable         access=use
```

Object type - workload application:
The following table gives the access keywords required to work with workload applications:

Table 60. Workload applications - access keywords

Activity			Access keywords required
Dynamic Workload Console	add	Add new workload applications templates to the database. Unlock access is needed to use the `;unlock` attribute.	add, unlock
	create	Create a workload application template in the database. Modify access is needed to use the `;lock` attribute.	display, modify
	delete	Delete a workload application template from the database.	delete
	display	Display a workload application template.	display
	list	List workload application templates in the database.	list
	lock	Lock workload application templates in the database.	modify
	modify	Modify a workload application template in the database.	add, modify
	new	Create a workload application template in the database.	add, modify
	rename	Rename workload application templates in the database. The user needs add access to the new object and delete and display access to the old object.	add, delete, display
	replace	Replace workload application templates in the database. Unlock access is needed to use the `;unlock` attribute.	add, modify, unlock
	unlock	Unlock workload application templates in the database.	unlock

Example: To allow a user to create and delete a workload application, specify:

```
wkldappl          access=add,delete
```

The *TWS_user* - special security file considerations

The TWS_user is a special user, and requires special consideration for the security file.

Required access for the *TWS_user* for workload service assurance

For any user to perform Workload service Assurance activities, the *TWS_user* must have *display*, *modify* and *list* access keywords assigned for all *job*, *schedule* and *cpu* objects.

New *TWS_user* in migrated Security file

If you change the *TWS_user* of your environment, for example, as you might do when performing a parallel upgrade, and then you migrate the Security file (to preserve your settings) you must set up the new *TWS_user* in the Security file in advance, with all its required access rights, before attempting to start IBM Workload Scheduler.

Update definitions for Windows domain *TWS_user* in the Security file after upgrade to version 9.4

Due to new support of the UPN Windows user, if you have Windows domain users that are defined in the logon fields as `domain\username`, after performing an upgrade to version 9.4, update the `Security` file before starting the IBM Workload Scheduler instance. Insert the escape character '\' before the '\' character in the `domain\username` value.

For example, if you use the MYDOMAIN\user1 value in the logon field, after the upgrade, in the Security file you must update the line in following way:

```
..............
logon=MYDOMAIN\\user1
..............
```

Sample security file

This section contains a sample security file divided into sections for each different class of user.

Note that the order of definitions is from most to least-specific. Because of the order, *TWS_users* and **root** users are matched first, followed by users in the **sys** group, and then users in the **mis** group. All other users are matched with the last definition, which is the least specific.

TWS_users and root users logged in on the master domain manager

user mastersm cpu=$master + logon=*TWS_user*,root

```
###########################################################
#     Sample Security File
###########################################################
# APPLIES TO TWS_users AND ROOT USERS LOGGED IN ON THE
# MASTER DOMAIN MANAGER.
user mastersm  cpu=$master + logon=TWS_user,root
begin
#  OBJECT     ATTRIBUTES        ACCESS CAPABILITIES
# ----------  ------------      ----------------------
job           cpu=@             access=@
schedule                        access=@
resource                        access=@
prompt                          access=@
file                            access=@
calendar                        access=@
cpu           cpu=@             access=@
parameter     name=@ ~ name=r@  access=@
userobj       cpu=@ + logon=@   access=@
eventrule     name=@            access=add,delete,display,modify,list,unlock
action        provider=@        access=display,submit,use,list
event         provider=@        access=use
report        name=@            access=display
runcygrp      name=@            access=add,delete,display,modify,use,list,unlock
vartable      name=a@,$default  access=add,delete,display,modify,use,list,unlock
wkldappl      name=@            access=add,delete,display,modify,list,unlock
lob           name=@            access=use
end
```

This user definition applies to GUI and CLI access for *TWS_users* and **root** users logged into a master domain manager. They are given unrestricted access to all objects, except parameters that have names beginning with **r**. Access to the **r** parameters is given only to users in the **mis** group. They are the only ones who can generate all kinds of plans and who can create, update, and delete event rule definitions.

All users have access to all variable tables beginning with "a" and to the default table, irrespective of the default variable table name.

TWS_users and root users logged in on any domain manager (other than the master)

user testerlondon cpu=$manager + logon=*TWS_user*,root

```
###########################################################
#     Sample Security File
###########################################################
# APPLIES TO TWS_users AND ROOT USERS LOGGED IN ON ANY
# DOMAIN MANAGER.
user testerlondon  cpu=$manager + logon=TWS_user,root
begin
#  OBJECT     ATTRIBUTES      ACCESS CAPABILITIES
# ----------  ------------    ---------------------
job           cpu=@           access=add,delete,display
schedule                      access=add,delete,display
resource                      access=@
prompt                        access=@
file          name=prodsked   access=build, display
file          name=trialsked  access=build, display
calendar                      access=@
cpu           cpu=@           access=@
parameter  name=@ ~ name=v@   access=@
userobj    cpu=@ + logon=@    access=@
eventrule       name=@        access=add,delete,display,modify,list,unlock
action     provider=@         access=display,submit,use,list
event      provider=@         access=use
report     name=@             access=display
runcygrp      name=@            access=add,delete,display,modify,use,list,unlock
vartable   name=a@,$default   access=add,delete,display,modify,use,list,unlock
wkldappl      name=@            access=add,delete,display,modify,list,unlock
lob           name=@            access=use
end
```

This user definition applies to GUI and CLI access for *TWS_users* and **root** users logged into any domain manager other than the master. They are given unrestricted access to all objects, except parameters that have names beginning with **v**, and jobs and jobs streams to which they have limited access. They can generate all types of plans and can create, update, and delete event rule definitions.

All users have access to all variable tables beginning with "a" and to the default table, irrespective of the default variable table name.

TWS_users and root users logged in on any workstation other than any domain manager

user sm ~CPU=$MANAGER logon=*TWS_user*,root

```
###########################################################
# APPLIES TO TWS_users AND ROOT USERS LOGGED IN ON ANY
# WORKSTATION OTHER THAN THE MASTER DOMAIN MANAGER.
user sm  logon=TWS_user,root
begin
#  OBJECT     ATTRIBUTES       ACCESS CAPABILITIES
# ----------  ------------     ---------------------
job           cpu=$thiscpu     access=@
schedule      cpu=$thiscpu     access=@
resource      cpu=$thiscpu     access=@
prompt                         access=@
calendar                       access=@
cpu           cpu=$thiscpu     access=@
parameter     cpu=$thiscpu
              ~ name=r@        access=@
action        provider=@       access=display,submit,use,list
event         provider=@       access=use
report        name=RUNHIST,RUNSTATS  access=display
runcygrp      name=@            access=add,delete,display,modify,use,list,unlock
file          name=globalopts  access=display
lob           name=@            access=use
end
```

This user definition applies to *TWS_users* and **root** users to whom definition (1) does not apply, which are those who are logged in on any workstation other than the master domain manager or any other domain manager. They are given unrestricted access to all objects on their login workstation. Note that prompts, files, and calendars are global in nature and are not associated with a workstation.

They can use event rules, but are not allowed to create, update, or delete event rule definitions.

Users logged into the *sys* group on the master domain manager

user masterop cpu=$master + group=sys

```
#########################################################
# APPLIES TO USERS LOGGED INTO THE SYS GROUP ON THE
# MASTER DOMAIN MANAGER.
user masterop  cpu=$master + group=sys
begin
#  OBJECT     ATTRIBUTES       ACCESS CAPABILITIES
# ----------  ------------     ----------------------
job           cpu=@
              + logon="TWS_domain\TWS_user" access=@
job           cpu=@
              + logon=root     access=adddep,altpri,cancel,
                                      confirm,deldep,release,
                                      reply,rerun,submit,use
job           cpu=@
              + logon=@
              ~ logon=root     access=add,adddep,altpri,
                                      cancel,confirm,
                                      deldep,release,reply,
                                      rerun,submit,use
schedule      cpu=$thiscpu     access=@
schedule      cpu=@            access=adddep,altpri,cancel,
                                      deldep,limit,release,
                                      submit
resource                       access=add,display,
                                      resource,use
file          name=globalopts  access=display
file          name=prodsked    access=display
file          name=symphony    access=display
file          name=trialsked   access=build, display
calendar                       access=display,use
cpu           cpu=@            access=@
parameter     name=@ ~ name=r@ access=@
report        name=RUNHIST,RUNSTATS  access=display
wkldappl      name=@            access=add,delete,display,modify,list,unlock
lob           name=@            access=use
end
```

This user definition applies to users logged into the **sys** group on the master domain manager. They are given a unique set of access capabilities. Multiple object statements are used to give these users specific types of access to different sets of objects. For example, there are three job statements:

- The first job statement permits unrestricted access to jobs that run on any workstation (@) under the user's name (*TWS_domain\TWS_user*).
- The second job statement permits specific types of access to jobs that run on any workstation and that run as **root**.
- The third job statement permits specific types of access to jobs that run on any workstation. Jobs that run as root are excluded.

They are the only users defined on the master domain manager, different from maestro or root, who can generate trial and forecast plans.

Users logged into the *sys* group on any workstation other than the master domain manager

user op ~cpu=$master group=sys

```
#########################################################
# APPLIES TO USERS LOGGED INTO THE SYS GROUP ON ANY
# WORKSTATION OTHER THAN THE MASTER DOMAIN MANAGER
user op  group=sys
begin
#  OBJECT     ATTRIBUTES       ACCESS CAPABILITIES
# ----------  ------------     ---------------------
job           cpu=$thiscpu
              + logon=@        access=@
job           cpu=$thiscpu
              + logon=root     access=adddep,altpri,cancel,
                                      confirm,deldep,release,
                                      reply,rerun,submit,use
job           cpu=$thiscpu
              ~ logon=root     access=adddep,altpri,cancel,
                                      confirm,deldep,release,
                                      reply,rerun,submit,use
schedule      cpu=$thiscpu     access=@
resource                       access=add,display,resource,use
runcygrp      name=@            access=add,delete,display,modify,use,list,unlock
prompt                         access=add,display,reply,use
calendar                       access=use
cpu           cpu=$thiscpu     access=console,fence,limit,
                                      link,start,stop,unlink
parameter     name=@ ~ name=r@ access=@

wkldappl      name=@            access=add,delete,display,modify,list,unlock
lob           name=@            access=use
end

#########################################################
```

This user definition applies to **sys** group users to whom definition (3) does not apply, which are those who are logged in on any workstation other than the master domain manager. They are given a set of access capabilities similar to those in definition (3). The exception is that access is restricted to objects on the user's login workstation (**$thiscpu**).

Users logged into the *mis* group on any workstation

user misusers group=mis

```
#########################################################
# APPLIES TO USERS LOGGED INTO THE MIS GROUP ON
# ANY WORKSTATION.
user misusers  cpu=@           group=mis
begin
#  OBJECT     ATTRIBUTES       ACCESS CAPABILITIES
# ----------  ------------     ---------------------
job           cpu=$thiscpu
              + logon=@        access=@
job           cpu=$thiscpu
              + logon=@
              ~ logon=root     access=submit,use
schedule      cpu=$thiscpu     access=add,submit,
                                      modify,display
cpu           cpu=@ + type=agent,s-agent,fta
                               access=console,fence,limit,
                                      link,start,stop,unlink
parameter     name=r@          access=@
parameter     name=@           access=display
```

```
runcygrp      name=@            access=add,delete,display,modify,use,list,unlock
end
#########################################################
```

This user definition applies to users logged into the **mis** group on any workstation. They are given a limited set of access capabilities to fault-tolerant, standard, and dynamic agents. Resources, prompts, files, calendars, and workstations are omitted, which prevents access to these objects. These users are given unrestricted access to parameters with names that begin with **r**, but can only display other parameters.

Users logged into multiple groups [continue keyword]

This is an example of a security file where the `continue` keyword is used. This kind of security file allows a user to inherit authorization from multiple *stanzas*. The user gets the accesses for the first matching entry of each *stanza* that matches the user definition.

user misusers cpu@ group=mis

```
#########################################################
# User misusers USER DEFINITION APPLIES TO USERS LOGGED IN TO
# THE MIS GROUP ON ANY WORKSTATION.
#
# User dbusers USER DEFINITION APPLIES TO USERS LOGGED IN TO
# THE DB GROUP ON ANY WORKSTATION.
#
# User default USER DEFINITION APPLIES TO ALL USERS.
#

user misusers  cpu=@           group=mis
begin
#  OBJECT     ATTRIBUTES       ACCESS CAPABILITIES
# ----------  ------------     ----------------------
job           cpu=@ + name=mis@
                               access=@
schedule      name=mis@        access=@
parameter     name=mis@        access=@
continue

user dbusers  cpu=@            group=db
begin
#  OBJECT     ATTRIBUTES       ACCESS CAPABILITIES
# ----------  ------------     ----------------------
job           cpu=@ + name=db_@
                               access=@
schedule      name=db_@        access=@
parameter     name=db_@        access=@
continue

user default cpu=@ + logon=@
begin
#  OBJECT     ATTRIBUTES       ACCESS CAPABILITIES
# ----------  ------------     ----------------------
parameter     name=@           access=display
end


#########################################################
```

Users that belong only to the *mis* group get access to all objects that have a name starting with the *mis* prefix, as specified in the `user misusers` user definition. In addition, the `user default` user definition gives them display access to all parameters.

Users that belong only to the *db* group get access to all objects that have a name starting with the *db_* prefix, as specified in the `user dbusers` user definition. In addition, the `user default` user definition gives them display access to all parameters.

Users that belong to both the *mis* and the *db* groups get access to the objects that have a name starting with the *mis* prefix and to the objects that have a name starting with the *db_* prefix, as specified in the `user misusers` and in the `user dbusers` user definitions. In addition, the `user default` user definition gives them display access to all parameters.

You must order definitions from most specific to least specific. The `user default` user definition gives generic accesses, and must be therefore specified at the end of the file.

All other users logged in on any workstation

user default cpu=@ + logon=@

```
############################################################
# APPLIES TO ALL OTHER USERS LOGGED IN ON ANY
# WORKSTATION.
user default  cpu=@ + logon=@
begin
#  OBJECT     ATTRIBUTES       ACCESS CAPABILITIES
# ----------  ------------     ----------------------
job           cpu=@             access=@
schedule                        access=@
resource                        access=@
prompt                          access=@
file                            access=@
calendar                        access=@
cpu           cpu=@             access=@
parameter     name=@ ~ name=r@  access=@
userobj       cpu=@ + logon=@   access=@
eventrule     name=@            access=add,delete,display,modify,list,unlock
action        provider=@        access=display,submit,use,list
event         provider=@        access=use
report        name=@            access=display
runcygrp      name=@            access=add,delete,display,modify,use,list,unlock
vartable      name=a@,$default  access=add,delete,display,modify,use,list,unlock
wkldappl      name=@            access=add,delete,display,modify,list,unlock
lob           name=@            access=use
end
############################################################
```

They are given unrestricted access to all objects, except parameters that have names beginning with **r**. They are the only ones who can generate all kinds of plans and who can create, update, and delete event rule definitions. All users have access to all variable tables beginning with "a" and to the default table, irrespective of the default variable table name.

All domain1.com windows users logged in on any workstation

user cpu=@ + logon =@\@domain1.com

```
############################################################
# APPLIES TO ALL OTHER USERS IN THE 'domain1.com' INTERNET DOMAIN LOGGED IN ON ANY
# WORKSTATION.
user default  cpu=@ + logon=@\@domain1.com
begin
#  OBJECT     ATTRIBUTES       ACCESS CAPABILITIES
# ----------  ------------     ----------------------
job           cpu=@ + logon =a@\@domain1.com  access=display
job           cpu=@             access=@
```

```
schedule                        access=@
resource                        access=@
prompt                          access=@
file                            access=@
calendar                        access=@
cpu           cpu=@             access=@
parameter     name=@ ~ name=r@  access=@
userobj       cpu=@ + logon=@   access=@
eventrule     name=@            access=add,delete,display,modify,list,unlock
action        provider=@        access=display,submit,use,list
event         provider=@        access=use
report        name=@            access=display
runcygrp      name=@            access=add,delete,display,modify,use,list,unlock
vartable      name=g@,$default  access=add,delete,display,modify,use,list,unlock
wkldappl      name=@            access=add,delete,display,modify,list,unlock
lob           name=@            access=use
end
###########################################################
```

Windows Users in `domain1.com` which name begins with 'a' can display only jobs and can manage parameters which name not beginning with **r**. All others `domain1.com` windows user that is logged in on any workstation are given unrestricted access to all objects, except parameters that have names beginning with **r**. They are the only ones who can generate all kinds of plans and who can create, update, and delete event rule definitions. All users have access to all variable tables beginning with "g" and to the default table, irrespective of the default variable table name.

All MYWINDOM windows users logged in on any workstation

user default cpu=@ + logon=MYWINDOM\\@

```
###########################################################
# APPLIES TO ALL "MYWINDOM" WINDOWS USERS LOGGED IN ON ANY
# WORKSTATION.
user default  cpu=@ +  logon=MYWINDOM\\@
begin
#  OBJECT     ATTRIBUTES       ACCESS CAPABILITIES
# ----------  ------------     ----------------------
job           cpu=@             access=@
schedule                        access=@
resource                        access=@
prompt                          access=@
file                            access=@
calendar                        access=@
cpu           cpu=@             access=@
parameter     name=@            access=@
userjob       cpu=@ + logon =MYWINDOM\\r@  access=display
userobj       cpu=@ + logon=@   access=@
eventrule     name=@            access=add,delete,display,modify,list,unlock
action        provider=@        access=display,submit,use,list
event         provider=@        access=use
report        name=@            access=display
runcygrp      name=@            access=add,delete,display,modify,use,list,unlock
vartable      name=g@,$default  access=add,delete,display,modify,use,list,unlock
wkldappl      name=@            access=add,delete,display,modify,list,unlock
lob           name=@            access=use
end
###########################################################
```

Windows Users in `MYWINDOM` which name begins with 'r' can display only userjobs. All others `MYWINDOM` windows user that is logged in on any workstation are given unrestricted access to all objects. They are the only ones who can generate all kinds of plans and who can create, update, and delete event rule definitions. All users

have access to all variable tables beginning with "g" and to the default table, irrespective of the default variable table name.

Note: Starting with version 9.2, due to support of the Windows users in User Principal Name (UPN) format, you have to specify the windows domain users in a different way in the Security file. In the same example for the previous version you have the following syntax:

```
user default  cpu=@ +  logon=MYWINDOM\@
.....................................
userjob       cpu=@ + logon =MYWINDOM\r@  access=display
```

Security file on the master domain manager to install fix packs or upgrade fault-tolerant agents and dynamic agents

user MAESTRO CPU=@+LOGON=tws93user,Administrator

```
###########################################################
# APPLIES TO tws93user and Administrator LOGGED IN ON ANY WORKSTATIONS.
###########################################################
USER MAESTRO
  CPU=@+LOGON=tws93user,Administrator
BEGIN
  USEROBJ  CPU=@  ACCESS=ADD,DELETE,DISPLAY,MODIFY,USE,ALTPASS,LIST,UNLOCK
  JOB  CPU=@  ACCESS=ADD,ADDDEP,ALTPRI,CANCEL,CONFIRM,DELDEP,DELETE,DISPLAY,
                     KILL,MODIFY,RELEASE,REPLY,RERUN,SUBMIT,USE,LIST,UNLOCK,
SUBMITDB,RUN
  SCHEDULE  CPU=@  ACCESS=ADD,ADDDEP,ALTPRI,CANCEL,DELDEP,DELETE,DISPLAY,LIMIT,
                           MODIFY,RELEASE,REPLY,SUBMIT,LIST,UNLOCK
  RESOURCE  CPU=@  ACCESS=ADD,DELETE,DISPLAY,MODIFY,RESOURCE,USE,LIST,UNLOCK
  PROMPT    ACCESS=ADD,DELETE,DISPLAY,MODIFY,REPLY,USE,LIST,UNLOCK
  FILE  NAME=@  ACCESS=BUILD,DELETE,DISPLAY,MODIFY,UNLOCK
  CPU  CPU=@  ACCESS=ADD,CONSOLE,DELETE,DISPLAY,FENCE,LIMIT,LINK,
                     MODIFY,SHUTDOWN,START,STOP,UNLINK,LIST,UNLOCK,RUN,
RESETFTA,MANAGE
  PARAMETER  CPU=@  ACCESS=ADD,DELETE,DISPLAY,MODIFY,LIST,UNLOCK
  CALENDAR    ACCESS=ADD,DELETE,DISPLAY,MODIFY,USE,LIST,UNLOCK
  REPORT  NAME=@  ACCESS=DISPLAY
  EVENTRULE  NAME=@  ACCESS=ADD,DELETE,DISPLAY,MODIFY,LIST,UNLOCK
  ACTION  PROVIDER=@  ACCESS=DISPLAY,SUBMIT,USE,LIST
  EVENT  PROVIDER=@  ACCESS=USE
  VARTABLE  NAME=@  ACCESS=ADD,DELETE,DISPLAY,MODIFY,USE,LIST,UNLOCK
  WKLDAPPL  NAME=@  ACCESS=ADD,DELETE,DISPLAY,MODIFY,LIST,UNLOCK
  RUNCYGRP  NAME=@  ACCESS=ADD,DELETE,DISPLAY,MODIFY,USE,LIST,UNLOCK
LOB        NAME=@  ACCESS=USE
END
###########################################################################
```

The default MAESTRO definition applies to Dynamic Workload Console and CLI access for `tws93user` and `Administrator` users logged into any workstation in the network. They can install a fix pack or upgrade to a later version fault-tolerant agents and dynamic agents in the network simultaneously.

For more information about this feature, see the section about centralized agent update in *Planning and Installation Guide*.

Chapter 5. Configuring authentication

This section describes how to configure authentication using, amongst other methods, the popular LDAP (Lightweight Directory Access Protocol). It is divided into these main topics:

Where to configure authentication

Authentication must be configured for each WebSphere Application Server profile, following these rules:

To authenticate command-line users

For users of the command-line, the command-line client, and the command-line as clients connected to the master domain manager using HTTP or HTTPS, the same authentication method must be configured for the following components:

- Master domain manager
- Backup master domain manager

To authenticate Dynamic Workload Console

The Dynamic Workload Console is not installed on the same instance as the master domain manager, authentication must be configured separately for theDynamic Workload Console and the master domain manager.

To authenticate z/OS connector users

The z/OS connector is always installed on the same instance as the Dynamic Workload Console. You do not need to separately configure authentication for it.

To authenticate dynamic domain manager users

The same authentication method must be configured for each dynamic domain manager and its corresponding backup dynamic domain manager. This authentication method does not need to be the same as that used for the master domain manager.

Available configurations

On installation, all IBM Workload Scheduler components that use the WebSphere Application Server are configured for authentication in VMM (Virtual Member Manager) mode. This creates a *Federated User Registry*, in which you can choose to use one or more of the following authentication systems:

- Local operating system - the default authentication system at installation on Windows operating systems
- Pluggable Authentication Module (PAM) - the default authentication system at installation on UNIX, Linux, and AIX® operating systems. `Custom` is the default authentication system at installation on UNIX and Linux operating systems, and on AIX operating systems, it is called `CustomPAM`, to differentiate it from the other authentication system supported on AIX called, CustomLAM.
- Loadable Authentication Module (LAM) - `CustomLAM` is an authentication and identification mechanism for AIX operating systems only.
- LDAP
- File Registry

Note: On AIX, the CustomPAM authentication system is mutually exclusive with both the CustomLAM and LocalOS authentication system choices.

If you want to use local OS as the authentication method for the Dynamic Workload Console on UNIX operating systems, perform the steps in "Configuring the Dynamic Workload Console to use the local OS or PAM authentication method" on page 110.

If you choose to enable LDAP, you can use one of the following servers, for which sample configuration templates are supplied in this documentation:

- IBM Tivoli Directory Server
- Sun Java Director Server
- Microsoft Windows Active Directory
- z/OS Integrated Security Services LDAP Server

How to configure authentication

About this task

LDAP can be configured in either of these ways:

Using the WebSphere Administrative Console

For each WebSphere Application Server profile where you want to modify the default authentication configuration, open theWebSphere Administrative Console and select to configure Global Security. You choose and configure the authentication mechanism or mechanisms you use in your environment.

See "Configuring authentication using the WebSphere Administrative Console" on page 248 for a full description.

Manually, using the WebSphere Application Server tools supplied with the product

For each WebSphere Application Server profile where you want to modify the default authentication configuration, run a script called `showSecurityProperties` to create a template containing the current security configuration. You modify this template by adding and amending

the properties that define the authentication mechanism or mechanisms you use in your environment. Finally, you run a script called `changeSecurityProperties` to update the WebSphere Application Server security configuration.

See "Configuring authentication using the WebSphere Application Server tools" on page 251 for a full description.

A typical configuration scenario

About this task

In a complex environment, you might want to use the following scenario to configure your chosen authentication mechanism across your workload scheduling environment:

1. Use the `changeSecurityProperties` script to configure the mechanism on one instance of WebSphere Application Server, for example that installed with the master domain manager.
2. Test that you can log in using the configured authentication with several user IDs.
3. On that instance run the `showSecurityProperties` script and save the output to create a text template file containing the configuration. `showSecurityProperties` extracts *only* those configurations that have been created by using the `changeSecurityProperties`.
4. On each of these systems where you want to configure authentication
   - Copy the text template file created in the previous step.
   - Run `showSecurityProperties` and save the output file.
   - Merge this output file with the configuration template file.
   - Run `changeSecurityProperties` to update the WebSphere Application Server configuration.
   - Test that you can log in using the configured authentication with several user IDs.

Rules for using a Federated User Registry with IBM Workload Scheduler

This section describes the simple rules you must follow when configuring IBM Workload Scheduler to use a Federated User Registry:

No duplicate User IDs

You can define any number of user registries in a Federated User Registry. However, no user ID must be present in more than one registry (this prohibits using both Local OS and PAM as a joint authentication mechanism) and no user ID must be present twice in the same registry. Thus, if you configure multiple user registries it is because you have users in different non-inclusive groups that use different user registries and which need to access IBM Workload Scheduler.

Reserved registry IDs

The WebSphere Application Server tools use some specific IDs to recognize the registries and these are thus reserved keywords that you cannot use to create your own registries, whichever method you use to configure them:

twaLocalOS

Identifies the custom user registry bridge adapter configured for local operating system users

twaPAM
Identifies the custom user registry bridge adapter configured to use the Pluggable Authentication Module (PAM) with IBM Workload Scheduler – it is not available on Windows operating systems

twaLDAP
Identifies the user registry bridge configured for LDAP users

defaultWIMFileBasedRealm
Identifies the default WebSphere Application Server File Registry

Configuring authentication using the WebSphere Administrative Console

About this task

The WebSphere Administrative Console is the administrative user interface of the Dynamic Workload Console and it is installed automatically with every instance of the WebSphere Application Server.

Note: If you create the repository using the WebSphere Administrative Console, the `showSecurityProperties` wastool might not show data for the repository.

Use WebSphere Administrative Console to configure authentication as follows:

1. **Backup the configuration**

   Backup the WebSphere Application Server configuration using the command **`backupConfig`**.

2. **Access the WebSphere Administrative Console**

   To access the WebSphere Administrative Console, use one of the following URLs using the current WebSphere Application Server administration credentials:

   ```
   https://<Hostname>:<adminSecurePort>//console/
   ibm
   http://<Hostname>:<adminPort>/ibm/console/
   ```

   where:

   Hostname
   The fully qualified hostname or the IP address of the computer.

   adminSecurePort
   If you connect with HTTPS, supply the WebSphere Application Server Administration secure port, the default value of which is `31124`.

   adminPort
   If you connect with HTTP, supply the WebSphere Application Server Administration port, the default value of which is `31123`.

   Example

   ```
   https://mypc:31124/ibm/console/
   ```

3. **Login to the console**

   Log into the console using the WebSphere Application Server credentials. You supplied these when you installed the component on this system (they might have been modified since then).

4. **Navigate to the security section**

   Select **Security ▸ Global security**

5. **Configure your required authentication mechanism or mechanisms.**

   In the User account repository section you see the default **Federated repositories** option selected. Click the adjacent **Configure** button. Use the WebSphere Administrative Console to configure your authentication mechanism or mechanisms. When you modify the rows in the **Repositories in the realm** table, the value **InternalFileRepository** corresponding to the Repository Identifier column must not be deleted.

   For example, click **Add Base entry to Realm ...** > **Add Repository...** to add a new repository, such as LDAP.

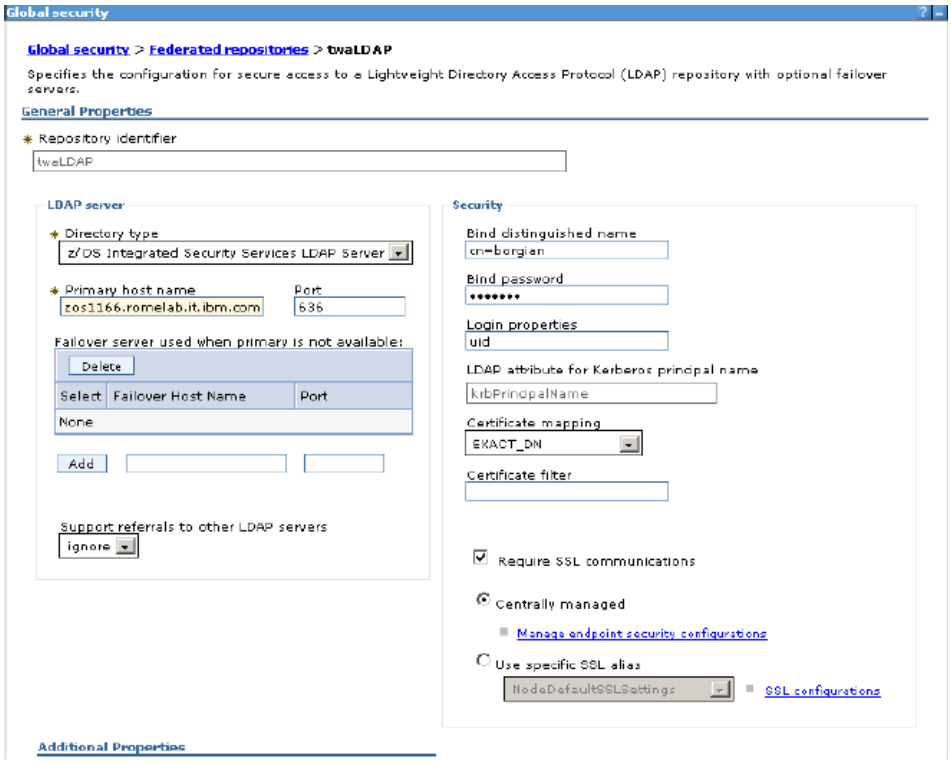
   Note: Do not delete the `twaPAM` entry from the repository until you have completed all the configuration steps.

   Use the built-in context-sensitive help to understand what information to supply in each field.

   In addition, all the key/value pairs output by the **`showSecurityProperties`** tool are documented in "Security properties: reference" on page 252. Each key/value pair corresponds to a field or concept expressed in the GUI of the WebSphere Administrative Console; the keys are mnemonic, to help you make the correspondence.

   Note: If you plan to configure the Dynamic Workload Console version 9.1 in Single Sign-On with IBM Workload Scheduler prior to 9.1, in the Global Security window, specify the same value in both the **Distinguished name of a base entry...** fields.

   See the following panel as example of a configuration with z/OS Integrate Security Service LDAP Server



6. **Save the modified configuration**

   Click **Save** to save the new configuration.

7. Then, you can modify the LDAP entity types for this repository. Under **Additional Properties** section, select **Supported entity types** > **Group**
8. In the Entity type field, enter the distinguished name of a base entry in the repository. This entry determines the default location in the repository where entities of this type are placed on write operations by user and group management.
9. Click **Apply** > **Save** to save current changes and return to previous panel.
10. You can specify the **Relative Distinguished Name** properties by entering the relative distinguished name (RDN™) properties for the specified entity type. Possible values are `cn` for **Group**, `uid` or `cn` for **PersonAccount**, and `o`, `ou`, `dc`, and `cn` for **OrgContainer**. Delimit multiple properties for the **OrgContainer** entity with a semicolon (;).
11. Click **OK** > **Save** to save the changes. While exiting you are be asked to set the base entry for this repository; the first name is mandatory and is a name of your choice that uniquely identify the repository in the federation. The second name is optional and depends on how the LDAP server is configured. See the following panel as an example:

12. **Restart the server**

    Stop the application server using the command **stopappserver**, as described in the *IBM Workload Scheduler: User's Guide and Reference*. To stop the server, use the original WebSphere administrator credentials.

    Restart the server using the command **startappserver**, as described in the *IBM Workload Scheduler: User's Guide and Reference*.
13. This step is applicable to Dynamic Workload Console only. Log into the Dynamic Workload Console :

    ```
    http://dynamic_workload_console_system:http_port/DASH_context_root
    https://dynamic_workload_console_system:https_port/DASH_context_root
    ```

    where,

    DASH_context_root

    It is the Dashboard Application Services Hub context root defined at installation time. The context root determines the URL of a deployed application and by default is identical with the application directory or archive structure. In this case, the default is `ibm/console`.

    Use the WebSphere Application Server credentials (the *original* administrative user) and assign the following roles to the primary administrative user name (the *new* administrative user).

    - Iscadmins
    - TDWBAdministrator

- TWSWEBUIAdministrator
- chartAdministrator

See "Configuring roles to access the Dynamic Workload Console" on page 111 for details.

Now, you can log into the Dynamic Workload Console as the *new* administrative user and, optionally, delete the `twaPAM` entry from the repository, if you do not need it anymore.

Configuring authentication using the WebSphere Application Server tools

About this task

When you install a IBM Workload Scheduler component that uses the WebSphere Application Server, you also install a set of WebSphere Application Server tools (also called *wastools*). For more general information about these tools, see "Application server utilities" on page 461.

You can use the application server tools to configure only the following LDAP servers:
- Microsoft Active Directory
- Oracle Java System Directory Server
- IBM Directory Server
- z/OS Integrated Security Services LDAP Server

For the other LDAP servers, follow the procedure described in "Configuring authentication using the WebSphere Administrative Console" on page 248.

For more information about LDAP server schema, see "LDAP server schema" on page 267.

To configure authentication, perform the following steps:

1. Log in to the Dynamic Workload Console with the current WebSphere Application Server administration credentials.
2. Back up the WebSphere Application Server configuration by using the command **backupConfig**.
3. For z/OS Integrated Security Services LDAP Server, import `cert.arm` in the WebSphere Application Serverby running IkeyMan Java tool the following paths, depending on the operating system:

   UNIX `/opt/IBM/<JazzSM_install_directory>/profile/bin/keyman.sh`

   Windows

   ```
   C:\Program Files\IBM\<JazzSM_install_directory>\profile\bin\
   ikeyman.bat
   ```

   The paths refer to the default location.
4. Dump your current security properties to a text file by using the command **showSecurityProperties <text_file>**
5. Customize the security properties by editing the *<text_file>*. See "Security properties: reference" on page 252.
6. Stop the server by using the command **stopappserver**, as described in the *IBM Workload Scheduler User's Guide and Reference*. To stop the server, use the original WebSphere administrator credentials.

7. Load the new properties by using the command **\<complete_path>\ changeSecurityProperties <text_file>**.
8. Restart the server by using the command **startappserver**, as described in the *IBM Workload Scheduler User's Guide and Reference*.

For an example of configurations of LDAP servers, see "Example configurations of LDAP servers" on page 264.

Security properties: reference

This section describes the important security properties in the file generated by the showSecurityProperties script. It is divided into *panels*:

Global Security Panel

Required panel.

```
#############################################################
Global Security Panel
#############################################################
enabled=true
enforceJava2Security=false
useDomainQualifiedUserNames=false
cacheTimeout=600
ltpaTimeOut=720
issuePermissionWarning=true
activeProtocol=CSI
useFIPS=false
activeAuthMechanism=LTPA
activeUserRegistry=LDAP LocalOS WIM Custom CustomLAM <repository_id>:
  <repository_base_name>
```

Note to users of previous versions of IBM Workload Scheduler: Nearly all of the properties are unchanged with respect to previous IBM Workload Scheduler releases.

The following property is new:

enabled=true | false
Specifies if application security is enabled (true) or not (false). The default is "true".

enforceJava2Security=false
Specify if Java 2 security is enabled (true). IBM Workload Scheduler does not support Java 2 security so this must be set to false (the default).

useDomainQualifiedUserNames=true | false
Specify if domain-qualified (realm-qualified) user names are to be used (true). If this is set to true, all user names in the Security file must be qualified with their domains. The default is false. Changing this value while using IBM Workload Scheduler could endanger your access to the product; if you need to do so discuss the best method with IBM Software Support.

cacheTimeout=<*seconds*>
Specifies the timeout value in seconds for the security cache. The security cache timeout can influence performance. The timeout setting specifies how often to refresh the security-related caches. Security information pertaining to beans, permissions, and credentials is cached. When the cache timeout expires, all cached information becomes invalid. Subsequent requests for the information result in a database lookup. Sometimes, acquiring the information requires invoking a Lightweight Directory Access

Protocol (LDAP)-bind or native authentication. Both invocations are relatively costly operations for performance. Determine the best trade off for the application, by looking at usage patterns and security needs for the site. The default security cache timeout value is 600 seconds. If you have a small number of users, it should be set higher than that, or if a large number of users, it should be set lower.

ltpaTimeout=<*seconds*>
Specifies the cache timeout for the LTPA data. The LTPA timeout value should not be set lower than the security cache timeout. The default is 720 seconds.

issuePermissionWarning=true
Specifies that during application deployment and application start, the security run time issues a warning if applications are granted any custom permissions (true). Custom permissions are permissions that are defined by the user applications, not Java API permissions. Java API permissions are permissions in the java.* and javax.* packages. For IBM Workload Scheduler leave the setting as "true".

activeProtocol=CSI
Specifies the active authentication protocol for Remote Method Invocation over the Internet Inter-ORB Protocol (RMI IIOP) requests, when security is enabled. For IBM Workload Scheduler leave the setting as "CSI".

useFIPS=true | false
Specify if the IBM Workload Scheduler network is FIPS compliant (true) and thus uses GSKit, for SSL or is not FIPS compliant (false), and uses OpenSSL. The default is false. See “FIPS compliance” on page 339 for more details.

activeAuthMechanism=LTPA
Specifies the active authentication mechanism. For IBM Workload Scheduler leave the setting as "LTPA".

activeUserRegistry=<*space_separated_list*>
Specifies a list of space-separated entries that identify the registries to enable. All the entries listed here will be enabled together in the VMM Federated User Registry. Allowed values are:

- `LocalOS`
- `Custom` (on UNIX and Linux operating systems), `CustomPAM` (on AIX operating systems)
- `CustomLAM`
- `LDAP`
- `WIM`
- <*REPOSITORY_ID*>:<*REPOSITORY_REALM_BASENAM*E>

  Use this if you have configured another repository using Integrated Solutions Console or any other mechanism other than the IBM Workload Scheduler WebSphere Application Server tools, and you want to enable such a repository either on its own or together with the default registries indicated above.

  For example, if you have created a repository with id "BluePages" and with a realm base name of "ibm.com®", you must specify:

  ```
  activeUserRegistry=BluePages:o=ibm.com <other_repository_ids>
  ```

Note: On AIX, the `CustomPAM` authentication system is mutually exclusive with both the `CustomLAM` and `LocalOS` values.

Federated repository panel

Required panel.

```
############################################################
Federated Repository Panel
############################################################
PrimaryAdminId=
UseRegistryServerId=
ServerID=
ServerPassword=
VMMRealm=TWSREALM
VMMRealmDelimiter=@
VMMIgnoreCase=true
```

Note to users of previous versions of IBM Workload Scheduler: This is a new panel.

PrimaryAdminId=<*name*>
Specifies the name of the user with administrative privileges which is defined in the repository, for example, `adminUser`. The user name is used to log on to the administrative console when administrative security is enabled. WebSphere Application Server requires an administrative user which is distinct from the server user identity so that administrative actions can be audited.

UseRegistryServerId=true | false
Specifies whether the server identity is to be generated automatically or supplied manually.

- If true, enables the application server to generate the server identity, which is recommended for environments that contain only WebSphere Application Server 6.1 or later nodes. Automatically generated server identities are not stored in a user repository.
- If false, requires that a user is specified as ServerID for internal process communication.

ServerID=<*name*>
Specifies a user identity in the repository that is used for internal process communication. Configurations that also contain WebSphere Application Server V6.0.x require a server user identity which is defined in the active user repository.

ServerPassword=<*password*>
Specifies the password that corresponds to the ServerID.

VMMRealm=<*name*>
Specifies the name of the realm. The default for this value is TWSREALM. Ensure this property is configured correctly for your environment, in order to allow communication between different servers and to improve speed.

VMMRealmDelimiter=<*value*>
Specifies the delimiter used to distinguish between user and realm when federating multiple repositories with different realms. The default value is "@".

VMMIgnoreCase=true | false
Specifies whether a case-insensitive authorization check is performed.

- If true, specifies that case sensitivity is not a consideration for authorization. Must be set to true when enabling LDAP repository with IBM Directory Server
- If false, the case of the user ID being authenticated will be used to match against the user IDs in the registry.

LDAP Panel

Complete this panel if configuring for an LDAP user registry.

```
############################################################
LDAP Panel
############################################################
LDAPServerType=IDS
LDAPHostName=
LDAPPort=389
LDAPBaseDN=
LDAPBaseDNEntry=
LDAPBindDN=
LDAPBindPassword=
LDAPloginProperties=
LDAPsearchTimeout=120
LDAPsslEnabled=false
LDAPsslConfig=
LDAPCertificateFilter=
LDAPCertificateMapMode=EXACT_DN
```

Note to users of previous versions of IBM Workload Scheduler: This is not a new panel, but is significantly different from previous versions.

LDAPServerType=<*name*>

Specifies the type of LDAP server to which you connect. If you use the IBM Tivoli Directory Server for z/OS, you must specify **IDS** . Allowed values are:

IDS IBM Tivoli Directory Server (the default LDAP value)

AD Microsoft Windows Active Directory

ZOSDS

z/OS Integrate Security Service LDAP Server

LDAPHostName=<*IP_address_or_hostname*>

Specifies the host name of the primary LDAP server. This host name is either an IP address or a domain name service (DNS) name.

LDAPPort=<*number*>

Specifies the LDAP server port. The default value is 389, which is not a Secure Sockets Layer (SSL) connection. Use port 636 for an SSL connection. For some LDAP servers, you can specify a different port for a non-SSL or SSL connection.

LDAPBaseDN=<*distinguished_name_list*>

Specifies the LDAP distinguished name (DN) of the base entry within the repository, which indicates the starting point for LDAP searches of the directory service. The entry and its descendants are mapped to the subtree that is identified by the "twaLDAP" base entry. If this field is left blank, then the subtree defaults to the root of the LDAP repository.

For example, for a user with a DN of `cn=John Doe , ou=Rochester, o=IBM, c=US`, specify that the `LDAPBaseDN` has any of the following options:

- `ou=Rochester, o=IBM, c=US`
- `o=IBM c=US`
- `c=US`

For authorization purposes, this field is case sensitive. This specification implies that if a token is received, for example, from another cell or Lotus® Domino®, the base DN in the server must match the base DN from the

other cell or Lotus Domino server exactly. If case sensitivity is not a consideration for authorization, enable the Ignore case for authorization option.

LDAPBaseDNEntry=<*distinguished_name_list*>
Specifies the LDAP distinguished name (DN) of a base entry that uniquely identifies the external repository in the realm. If multiple repositories are included in the realm, use this field to define an additional distinguished name (DN) that uniquely identifies this set of entries within the realm. If this field is left blank, then the default is:

LDAPBaseDN
For the first customization.

old value
For the succeeding customization. The default value is **o=twaLDAP**.

For example, repositories LDAP1 and LDAP2 might both use o=ibm, c=us as the base entry in the repository. Use the DN in this field to uniquely identify this set of entries in the realm. For example: o=ibm,c=us for LDAP1 and o=ibm2,c=us for LDAP2. The specified DN in this field maps to the LDAP DN of the base entry within the repository.

LDAPBindDN=<*name*>
Specifies the distinguished name (DN) for the application server to use when binding to the LDAP repository. If no name is specified, the application server binds anonymously. In most cases, `LDAPBindDN` and `LDAPBindPassword` are needed. However, when anonymous bind can satisfy all of the required functions, `LDAPBindDN` and `LDAPBindPassword` are not needed.

LDAPBindPassword=<*password*>
Specifies the password for the application server to use when binding to the LDAP repository.

LDAPloginProperties=<*login_token_list*>
Specifies the login tokens to use to log into the application server. This field takes multiple login tokens, delimited by a semicolon (;). For example, `uid;mail`. All login properties are searched during login. If multiple entries or no entries are found, an error is given. For example, if you specify the login properties as `uid;mail` and the login ID as Bob, the search filter searches for `uid=Bob` or `mail=Bob`. When the search returns a single entry, then authentication can proceed. Otherwise, an error is given.

If you supply more than one login token, the order you give to the login tokens is very important, because regardless of the token by which the user is authenticated, VMM sets the first property as the principal name. This principal name is then passed to IBM Workload Scheduler. For example, if you set the login properties to `cn;mail`, even if the user logs in with "mail", the principal name returned will be "cn"and the IBM Workload Scheduler security checks (in the security file, for example) are performed using the "cn" value for that user.

LDAPsearchTimeout=<*value*>
Specifies the timeout value in milliseconds for an LDAP server to respond before the request is aborted. A value of 0 specifies that no search time limit exists.

LDAPsslEnabled=true | false

Specifies whether secure socket communication is enabled to the LDAP server.

- If true, SSL is enabled, and the Secure Sockets Layer (SSL) settings for LDAP are used, if specified.
- If false, SSL is not enabled.

LDAPsslConfig=<*alias*>

Specifies the SSL configuration alias to use for LDAP outbound SSL communications. This option overrides the centrally managed configuration for the JNDI platform. The default value is `"DefaultNode/DefaultSSLSettings"`.

LDAPCertificateFilter=<*filter_specification*>

Specifies the filter certificate mapping property for the LDAP filter. The filter is used to map attributes in the client certificate to entries in the LDAP repository. If more than one LDAP entry matches the filter specification at run time, authentication fails because the result is an ambiguous match. The syntax or structure of this filter is:

<LDAP_attribute>=${<Client_certificate_attribute>}

For example, `uid=${SubjectCN}`.

The left side of the filter specification is an LDAP attribute that depends on the schema that your LDAP server is configured to use. The right side of the filter specification is one of the public attributes in your client certificate. The right side must begin with a dollar sign ($) and open bracket ({) and end with a close bracket (}). You can use any of the following certificate attribute values on the right side of the filter specification (the case of the strings is important):

- ${UniqueKey}
- ${PublicKey}
- ${PublicKey}
- ${Issuer}
- ${NotAfter}
- ${NotBefore}
- ${SerialNumber}
- ${SigAlgName}
- ${SigAlgOID}
- ${SigAlgParams}
- ${SubjectCN}
- ${Version}

LDAPCertificateMapMode=<*value*>

Specifies whether to map X.509 certificates into an LDAP directory by `EXACT_DN` or `CERTIFICATE_FILTER`.

Advanced LDAP Panel

Complete this panel if configuring for an LDAP user registry.

```
#############################################################
Advanced LDAP Panel
#############################################################
LDAPUserEntityType=PersonAccount
LDAPUserObjectClasses=
LDAPUserSearchBases=
LDAPUserSearchFilter=
LDAPUserRDNAttributes=
LDAPGroupEntityType=Group
```

```
LDAPGroupObjectClasses=
LDAPGroupSearchBases=
LDAPGroupSearchFilter=
LDAPGroupSearchFilter=
LDAPGroupRDNAttributes=
LDAPOrgContainerEntityType=OrgContainer
LLDAPOrgContainerObjectClasses=
LDAPOrgContainerSearchBases=
LDAPOrgContainerSearchFilter=
LDAPOrgContainerRDNAttributes=
LDAPGroupConfigName=ibm-allGroups
LDAPGroupConfigScope=all
LDAPGroupConfigMemberNames=member;uniqueMember
LDAPGroupConfigMemberClasses=groupOfNames;groupOfUniqueNames
LDAPGroupConfigMemberScopes=direct;direct
LDAPGroupConfigMemberDummies=uid=dummy;
```

Note to users of previous versions of IBM Workload Scheduler: It is not a new panel, but is different from previous versions.

LDAPUserEntityType=<*value***>**
Specifies the `PersonAccount` entity type name that is supported by the member repositories. By default, this value is "`PersonAccount`"

LDAPUserObjectClasses=<*entity_type_list***>**
Specifies the object classes that are mapped to the `PersonAccount` entity type. You can specify multiple values delimited by a semicolon (;) LDAP entries that contain one or more of the object classes belong to this entity type. You cannot map multiple entity types to the same LDAP object class.

LDAPUserSearchBases=<*search_base_list***>**
Specifies the search bases that are used to search the `PersonAccount` entity type. The search bases specified must be subtrees of the base entry in the repository.

For example, you can specify the following search bases, where `o=ibm,c=us` is the base entry in the repository:

- `o=ibm,c=us`
- `cn=users,o=ibm,c=us`
- `ou=austin,o=ibm,c=us`

In the preceding example, you cannot specify search bases `c=us` or `o=ibm,c=uk`.

Delimit multiple search bases with a semicolon (;). For example: `ou=austin,o=ibm,c=us;ou=raleigh,o=ibm,c=us`

LDAPUserSearchFilter=<*name***>**
Specifies the LDAP search filter that is used to search the `PersonAccount` entity type. If a search filter is not specified, the object classes and the relative distinguished name (RDN) properties are used to generate the search filter.

LDAPUserRDNAttributes=<*rdn_attributes_list***>**
Specifies the relative distinguished name (RDN) properties that are used to generate the search filter for the `PersonAccount` entity type. You can specify multiple values delimited by a semicolon (;). Specify this value in the form <*rdn_attribute_name*>:<*object_class*>. *object_class* is optional; if you specify it, this value is used by the **PersonAccount** entity type to map the corresponding *rdn_attribute_name*.

LDAPGroupEntityType=<*name***>**
Specifies the Group entity type name that is supported by the member repositories. By default, this value is **Group**.

LDAPGroupObjectClasses=<*object_classes_list***>**
Specifies the object classes that are mapped to the Group entity type. You can specify multiple values delimited by a semicolon (;) LDAP entries that contain one or more of the object classes belong to this entity type. You cannot map multiple entity types to the same LDAP object class.

LDAPGroupSearchBases=<*search_base_list***>**
Specifies the search bases that are used to search the Group entity type. The search bases specified must be subtrees of the base entry in the repository. See "LDAPUserSearchBases" for more information.

LDAPGroupSearchFilter=<*name***>**
Specifies the LDAP search filter that is used to search the Group entity type. If a search filter is not specified, the object classes and the relative distinguished name (RDN) properties are used to generate the search filter.

LDAPGroupRDNAttributes=<*rdn_attributes_list***>**
Specifies the relative distinguished name (RDN) properties that are used to generate the search filter for the **Group** entity type. You can specify multiple values delimited by a semicolon (;). Specify this value in the form *<rdn_attribute_name>*:*<object_class>*. *object_class* is optional; if you specify it, this value is used by the **Group** entity type to map the corresponding *rdn_attribute_name*.

LDAPOrgContainerEntityType=<*name***>**
Specifies the `OrgContainer` entity type name that is supported by the member repositories. By default, this value is "`OrgContainer`".

LDAPOrgContainerObjectClasses=<*object_classes_list***>**
Specifies the object classes that are mapped to the `OrgContainer` entity type. You can specify multiple values delimited by a semicolon (;) LDAP entries that contain one or more of the object classes belong to this entity type. You cannot map multiple entity types to the same LDAP object class.

LDAPOrgContainerSearchBases=<*search_base_list***>**
Specifies the search bases that are used to search the `OrgContainer` entity type. The search bases specified must be subtrees of the base entry in the repository. See "`LDAPUserSearchBases`" for more information.

LDAPOrgContainerSearchFilter=<*name***>**
Specifies the LDAP search filter that is used to search the `OrgContainer` entity type. If a search filter is not specified, the object classes and the relative distinguished name (RDN) properties are used to generate the search filter.

LDAPOrgContainerRDNAttributes=<*rdn_attributes_list***>**
Specifies the relative distinguished name (RDN) properties that are used to generate the search filter for the `OrgContainer` entity type. You can specify multiple values delimited by a semicolon (;). Specify this value in the form *rdn_attribute_name*:*<object_class>*. *object_class* is optional; if you specify it, this value is used by the **OrgContainer** entity type to map the corresponding *rdn_attribute_name*.

LDAPGroupConfigName=<*value***>**
Specifies the name of the group membership attribute. Only one

membership attribute can be defined for each LDAP repository. Every LDAP entry must have this attribute to indicate the groups to which this entry belongs.

For example, `memberOf` is the name of the membership attribute that is used in Active Directory. `Ibm-allGroups` is the name of the membership attribute that is used in IBM Tivoli Directory Server. The group membership attribute contains values that reference groups to which this entry belongs. If `User` belongs to `Group`, then the value of the `memberOf` attribute of `User` must contain the distinguished name of `Group`. If your LDAP server does not support the group membership attribute, then do not specify this attribute. The LDAP repository can look up groups by searching the group member attributes, though the performance might be slower.

LDAPGroupConfigScope=<*value*>
Specifies the scope of the group membership attribute. The default value is direct. For IBM Tivoli Directory Server the value to specify is "`all`". For Active Directory the value to specify is "`direct`" Allowed values:

direct The membership attribute contains direct groups only. Direct groups are the groups that contain the member.

For example, if `Group1` contains `Group2` and `Group2` contains `User1`, then `Group2` is a direct group of `User1`, but `Group1` is not a direct group of `User1`.

nested The membership attribute contains both direct groups and nested groups.

all The membership attribute contains direct groups, nested groups, and dynamic members.

LDAPGroupConfigMemberNames=<*names_list*>
Specifies the names of the "member attributes" in LDAP. You can specify more "member attributes" separating them with ";".

For example, `member` and `uniqueMember` are two commonly used names of member attributes. The member attribute is used to store the values that reference members that the group contains. For example, a group type with an object class `groupOfNames` has a member attribute named `member`; group type with object class `groupOfUniqueNames` has a member attribute named `uniqueMember`.

An LDAP repository supports multiple group types if multiple member attributes and their associated group object classes are specified.

LDAPGroupConfigMemberClasses=groupOfNames;groupOfUniqueNames
Specifies the object class of the group that uses these member attributes. If this field is not defined, this member attribute applies to all group object classes. You can specify more "member classes" separating them with ";".

If you specify more than one value in "`LDAPGroupConfigMemberNames`", then you can specify the class related to the specific member name defining the right value in the right position.

Example:

```
LDAPGroupConfigMemberNames=member;uniqueMember
LDAPGroupConfigMemberClasses=groupOfNames;groupOfUniqueNames
```

LDAPGroupConfigMemberScopes
Specifies the scope of the members attribute. You can specify more

"member scopes" separating them with ";". The default value is direct. If you specify more than one value in "LDAPGroupConfigMemberNames", then you can specify the scope related to the specific member name defining the right value in the right position. Allowed values:

direct The member attribute contains direct members only. Direct members are members that are directly contained by the group. For example, if Group1 contains Group2 and Group2 contains User1, then User1 is a direct member of Group2, but User1 is not a direct member of Group1.

nested The member attribute contains both direct members and nested members.

all The member attribute contains direct members, nested members, and dynamic members.

Example:

```
LDAPGroupConfigMemberNames=member;uniqueMember
LDAPGroupConfigMemberScopes=direct;all
```

LDAPGroupConfigMemberDummies=uid=dummy;

Indicates that if you create a group without specifying a member a dummy member is filled in to avoid creating an exception about missing a mandatory attribute. Allowed value is "uid=dummy" If you specify more than one value in "LDAPGroupConfigMemberNames", then you can specify the dummy member related to the specific member name defining the right value in the right position.

Example:

```
LDAPGroupConfigMemberNames=member;uniqueMember
LDAPGroupConfigMemberDummies=uid=dummy;
```

It means that only "member" have a "dummy member", while "uniqueMember" do not have a "dummy member" enabled.

SSL Panel

This panel is used to configure SSL and is not relevant to user authentication. All properties are unchanged with respect to the previous version.

J2C Authentication Data Panel

This panel is used to configure J2C Authentication and is not relevant to user authentication. All properties are unchanged with respect to the previous version.

ChangeSecurityProperties - output

The output of the ChangeSecurityProperties script contains messages that help you to understand if the configuration changes you made have been accepted.

The following example shows a sample output of the script:

```
--------------------------------------------------------------------------------
I: Using Property File: C:/TWS/wastools/FRESHI~1.TXT

I: Configuring Global Security ...
I: The LTPA LTPA has been found.
I: The LTPA timeout has been set to 720 minutes
I: Setting the authentication mechanism to (cells/DefaultNode|security.xml#LTPA_1)
I: Configuring SSL ...
I: Configuring LocalOS registry ...
I: twaLocalOS user registry bridge already exists
I: Configuring Advanced J2C Auth
```

```
I: twaLDAP LDAP user registry already exists
I: Configuring LDAP ...
I: Configuring Advanced LDAP ...
I: Enabling "LocalOS" user registry bridge
I: Enabling "LDAP" user registry bridge
I: The Active Authentication Mechanism is (cells/DefaultNode|security.xml#LTPA_1)
I: The Active User Registry is (cells/DefaultNode|security.xml#WIMUserRegistry_1)
   with base entries:
   o=twaLocalOS
   o=twaLDAP
I: The VMM useRegistryServerId property is "true"
I: The VMM ignoreCase property is "true"
I: The VMM realm is "TWSREALM"
I: Current LDAPServerType for user registry with id "twaLDAP" is "IDS"
I: The activeAuthmechanism is LTPA

I: Validation success. Configuration saved
-------------------------------------------------------------------------------
```

Each message begins with a letter indicating whether it is Informational (I), a Warning (W), or an Error (E).

Note:

1. If an error occurs, the configuration is not changed.
2. If a property is not supplied in the input file, the corresponding field in the embedded WebSphere Application Server is not updated.
3. If a password field is blank or "*****", the corresponding password in the embedded WebSphere Application Server is not updated.

Completing the configuration

About this task

After you have configured the WebSphere Application Server to use a new authentication configuration, whichever configuration method you used, you must also perform the following steps:

1. Create users and groups

Perform the following steps to create users and groups after you configured the new user registry. In this example, you use the Dynamic Workload Console but you can achieve the same result also by using the **composer**.

1. Log in to the WebSphere Application Server by using the modified WebSphere Application Server administrative user and password.
2. Assign the `TWSWEBUIAdministrator` role to the new administrative user.
3. Create new users and groups and assign roles to them, as explained in "Configuring roles to access the Dynamic Workload Console" on page 111.

2. Update the IBM Workload Scheduler security file

You need to update the IBM Workload Scheduler security file to allow users to access IBM Workload Scheduler objects (for more detailed information, see "Updating the security file" on page 204). The following example shows an updated security file, where the user `TEST_LDAP` has been added to the `USER MAESTRO` section:

```
USER MAESTRO
 CPU=@+LOGON=tws83,Administrator,administrator,TEST_LDAP
BEGIN
 USEROBJ CPU=@  ACCESS=ADD,DELETE,DISPLAY,MODIFY,ALTPASS,UNLOCK,LIST
```

```
 JOB  CPU=@  ACCESS=ADD,ADDDEP,ALTPRI,CANCEL,CONFIRM,DELDEP,DELETE,DISPLAY,KILL,
                  MODIFY,RELEASE,REPLY,RERUN,SUBMIT,USE,LIST,UNLOCK
 SCHEDULE CPU=@ ACCESS=ADD,ADDDEP,ALTPRI,CANCEL,DELDEP,DELETE,
        DISPLAY,LIMIT,MODIFY,RELEASE,REPLY,SUBMIT,LIST,UNLOCK
 RESOURCE CPU=@ ACCESS=ADD,DELETE,DISPLAY,MODIFY,RESOURCE,USE,LIST,UNLOCK
 PROMPT    ACCESS=ADD,DELETE,DISPLAY,MODIFY,REPLY,USE,LIST,UNLOCK
 FILE  NAME=@ ACCESS=CLEAN,DELETE,DISPLAY,MODIFY,UNLOCK
 CPU  CPU=@  ACCESS=ADD,CONSOLE,DELETE,DISPLAY,FENCE,LIMIT,LINK,MODIFY,
        SHUTDOWN,START,STOP,UNLINK,LIST,UNLOCK
 PARAMETER  CPU=@ ACCESS=ADD,DELETE,DISPLAY,MODIFY,UNLOCK,LIST
 CALENDAR    ACCESS=ADD,DELETE,DISPLAY,MODIFY,USE,UNLOCK,LIST
END
```

In this example, the **useDomainQualifiedUserNames** security property is set to **false** therefore the user name has been specified without the domain.

3. Update associated WebSphere Application Server properties

On both Windows and UNIX operating systems, after you modified the WebSphere Application Server to use LDAP, it is important that you change the SOAP client properties and revalidate the user credentials of the Windows services:

Update SOAP client properties

To update the SOAP client properties, use the **updateWAS.sh/.bat** script as follows. (For more detailed information, see "Application server - updating the SOAP properties after changing the WebSphere Application Server user or its password" on page 453).

```
updateWas.sh -user john.smith@domain.com -password zzzz
```

where **-user** and **-password** are set to the new logical user which is authorized to stop the WebSphere Application Server. The user must be defined in the user registry.

Update Windows services

On Windows, after the WebSphere Application Server was modified to use LDAP, you must update or revalidate the credentials of the Windows services by using the **updateWasService.bat** script as follows. (For detailed information, see"Application server - updating the Windows services after modifications" on page 452).

```
updateWasService -userid tws83 -password zzzz
–wasuser TEST_LDAP –waspassword xxxxxx
```

where **–userid** and **–password** are set to the operating system user ID and password of the user running the WebSphere Application Server process, and **–wasuser** and **–waspassword** are set to the new logical user authorized to stop the WebSphere Application Server. The **–wasuser** option must be defined in the user registry.

4. Propagate the changes

Propagate the changes you have made, as follows:

1. Update the USERNAME and PASSWORD fields in the `useropts` file on every command-line client that points to your workstation
2. Update the USERNAME and PASSWORD fields in the `useropts` file on every fault-tolerant agent in your environment that has an HTTP/HTTPS connection defined in localopts that points to your workstation. The HTTP/HTTPS connection is used to submit a predefined job or jobstream.

3. Update the USERNAME and PASSWORD fields in the engine connection parameters on every connected Dynamic Workload Console.

Note: To change the useropts file, change the USERNAME and type the new PASSWORD in plain text between double quotation marks. The password will be encrypted the first time you log in.

Example configurations of LDAP servers

Refer to this template also if you are using an IBM Tivoli Directory Server (ITDS) for z/OS LDAP server to access to information stored in RACF. Note that on the Integrated Solutions Console, LDAP users are queried only by the userid attribute. Check that an auxiliary class of **eperson** type and an uid attribute is added to the LDAP user ID.

Active Directory

```
################################################################
Global Security Panel
################################################################
enabled=true
enforceJava2Security=false
useDomainQualifiedUserNames=false
cacheTimeout=600
ltpaTimeOut=720
issuePermissionWarning=true
activeProtocol=CSI
useFIPS=false
activeAuthMechanism=LTPA
activeUserRegistry=WIM LocalOS LDAP

################################################################
Federated Repository Panel
################################################################
PrimaryAdminId=tws_admin
UseRegistryServerId=false
ServerID=
ServerPassword=
VMMRealm=myrealm
VMMRealmDelimiter=@
VMMIgnoreCase=true

################################################################
LDAP Panel
################################################################
LDAPServerType=AD
LDAPHostName=nc125088.romelab.it.ibm.com
LDAPPort=389
LDAPBaseDN=dc=test,dc=it
LDAPBaseDNEntry=dc=test,dc=it
LDAPBindDN=CN=ldap bind,DC=test,DC=it
LDAPBindPassword=*****
LDAPloginProperties=uid
LDAPsearchTimeout=120
LDAPsslEnabled=false
LDAPsslConfig=DefaultNode/DefaultSSLSettings
LDAPCertificateFilter=
LDAPCertificateMapMode=EXACT_DN
################################################################
Advanced LDAP Panel
################################################################
LDAPUserEntityType=PersonAccount
LDAPUserObjectClasses=user
LDAPUserSearchBases=
LDAPUserSearchFilter=(objectCategory=user)
```

```
LDAPUserRDNAttributes=sAMAccountName:user
LDAPGroupEntityType=Group
LDAPGroupObjectClasses=group
LDAPGroupSearchBases=
LDAPGroupSearchFilter=(objectCategory=group)
LDAPGroupRDNAttributes=cn:group
LDAPOrgContainerEntityType=OrgContainer
LDAPOrgContainerObjectClasses=organization;organizationalUnit
;domain;container
LDAPOrgContainerSearchBases=
LDAPOrgContainerSearchFilter=
LDAPOrgContainerRDNAttributes=ou:organizationalUnit;cn:container;
dc:domain;o:organization

LDAPGroupConfigName=memberOf
LDAPGroupConfigScope=direct
LDAPGroupConfigMemberNames=member
LDAPGroupConfigMemberClasses=groupOfNames
LDAPGroupConfigMemberScopes=direct
LDAPGroupConfigMemberDummies=
```

IBM Tivoli Directory Server

```
#################################################################
Global Security Panel
#################################################################
enabled=true
enforceJava2Security=false
useDomainQualifiedUserNames=false
cacheTimeout=600
ltpaTimeOut=720
issuePermissionWarning=true
activeProtocol=CSI
useFIPS=false
activeAuthMechanism=LTPA
activeUserRegistry= WIM LocalOS LDAP

#################################################################
Federated Repository Panel
#################################################################
PrimaryAdminId=tws_admin
UseRegistryServerId=false
ServerID=
ServerPassword=
VMMRealm=myrealm
VMMRealmDelimiter=@
VMMIgnoreCase=true
#################################################################
LDAP Panel
#################################################################
LDAPServerType=IDS
LDAPHostName=myhostname
LDAPPort=389
LDAPBaseDN=o=ibm.com
LDAPBindDN=
LDAPBindPassword=
LDAPloginProperties=mail;cn
LDAPsearchTimeout=120000
LDAPsslEnabled=false
LDAPsslConfig=
LDAPCertificateFilter=
LDAPCertificateMapMode=
#################################################################
Advanced LDAP Panel
#################################################################
LDAPUserEntityType=PersonAccount
LDAPUserObjectClasses=user;ePerson
LDAPUserSearchBases=
```

```
LDAPUserSearchFilter=(objectclass=ePerson)
LDAPUserRDNAttributes=mail:ePerson;cn:ePerson
LDAPGroupEntityType=Group
LDAPGroupObjectClasses=group;groupOfUniqueNames
LDAPGroupSearchBases=
LDAPGroupSearchFilter=(&(ou=memberlist)(ou=ibmgroups)
(o=ibm.com)(objectclass=groupOfUniqueNames))
LDAPGroupRDNAttributes=cn:groupOfUniqueNames
LDAPOrgContainerEntityType=OrgContainer
LDAPOrgContainerObjectClasses=organization;
organizationalUnit;domain;container
LDAPOrgContainerSearchBases=
LDAPOrgContainerSearchFilter=
LDAPOrgContainerRDNAttributes=ou:organizationalUnit;
cn:container;dc:domain;o:organization
LDAPGroupConfigName=ibm-allGroups
LDAPGroupConfigScope=all
LDAPGroupConfigMemberNames=uniqueMember
LDAPGroupConfigMemberClasses=groupOfUniqueNames
LDAPGroupConfigMemberScopes=direct
LDAPGroupConfigMemberDummies=
```

z/OS Integrate Security Service LDAP Server

```
#################################################################
Global Security Panel
#################################################################
enabled=true
enforceJava2Security=false
useDomainQualifiedUserNames=false
cacheTimeout=600
ltpaTimeOut=720
issuePermissionWarning=true
activeProtocol=CSI
useFIPS=false
activeAuthMechanism=LTPA
activeUserRegistry=LocalOS WIM LDAP
#################################################################
Federated Repository Panel
#################################################################
PrimaryAdminId=borgian
UseRegistryServerId=true
ServerID=tws86
ServerPassword=*****
VMMRealm=zos1166.romelab.it.ibm.com:636
VMMRealmDelimiter=@
VMMIgnoreCase=false
#################################################################
LDAP Panel
#################################################################
LDAPServerType=ZOSDS
LDAPHostName=zos1166.romelab.it.ibm.com
LDAPPort=636
LDAPBaseDN=ou=TIVOLI,o=IBM
LDAPBaseDNEntry=ou=TIVOLI,o=IBM
LDAPBindDN=cn=borgian
LDAPBindPassword=*****
LDAPloginProperties=uid
LDAPsearchTimeout=
LDAPsslEnabled=true
LDAPsslConfig=
LDAPCertificateFilter=
LDAPCertificateMapMode=exactdn
#################################################################
Advanced LDAP Panel
#################################################################
LDAPUserEntityType=PersonAccount
LDAPUserObjectClasses=eperson
```

```
LDAPUserSearchBases=ou=TIVOLI,o=IBM,
LDAPUserSearchFilter=
LDAPUserRDNAttributes=
LDAPGroupEntityType=Group
LDAPGroupObjectClasses=groupOfNames
LDAPGroupSearchBases=
LDAPGroupSearchFilter=
LDAPGroupRDNAttributes=
LDAPOrgContainerEntityType=OrgContainer
LDAPOrgContainerObjectClasses=organization;organizationalUnit;
domain;container
LDAPOrgContainerSearchBases=
LDAPOrgContainerSearchFilter=
LDAPOrgContainerRDNAttributes=ou:organizationalUnit;cn:
container;dc:domain;o:organization
LDAPGroupConfigName=
LDAPGroupConfigScope=
LDAPGroupConfigMemberNames=member
LDAPGroupConfigMemberClasses=groupOfNames
LDAPGroupConfigMemberScopes=direct
LDAPGroupConfigMemberDummies=uid=dummy
#################################################################
SSL Panel
#################################################################
alias=DefaultSSLSettings
keyFileName=${USER_INSTALL_ROOT}/etc/TWSServerKeyFile.jks
keyFilePassword=*****
keyFileFormat=JKS
trustFileName=${USER_INSTALL_ROOT}/etc/TWSServerTrustFile.jks
trustFilePassword=*****
trustFileFormat=JKS
clientAuthentication=false
securityLevel=HIGH
enableCryptoHardwareSupport=false
#################################################################
J2C Authentication Data Panel
#################################################################
j2cAlias=twsj2c
j2cUserid=db2admin
j2cPassword=*****
j2cDescription=TWS authentication data entry for data source
```

LDAP server schema

About this task

When defining the schema in the LDAP server, consider that the Dynamic Workload Console 8.6 is based on Dashboard Application Services Hub whose queries to the LDAP server assume that the users have the **uid** attribute defined. LDAP users are queried only by the `userid` attribute. When users are imported into LDAP using an LDAP Data Interchange Format (LDIF) file, an auxiliary class of type `eperson` and an `uid` attribute is added to the LDAP user ID.

For more information, see: the section about configuring an external LDAP repository: http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp?.

Therefore the LDAP server schema must contain the **uid** attribute and the **object class** must be `eperson` (the `person` object class used by the default schema does not support such an attribute). Moreover to comply with IDS for z/OS manual the **useNativeAuth** was set to all.

Example

Example of LDAP users defined:

```
LDAP Search is started ....
 Host = zos1166.MyUnit.es.MyOrg.com
 Port = 636
 Connection Type = SSL
 Timeout = 10 seconds
STEP 1 => Performing LDAP-SSL initialization
LDAP SSL initialization completed
STEP 2 => Connecting to LDAP server using the given credentials...
LDAP bind completed successfully.
STEP 3 => Searching on the server ...
------------------------------------------------------------------
Enumerating attributes for DN : cn=John Doe, ou=MyUnit, o=MyOrg
cn = John Doe
sn = BORGIAN
objectclass = organizationalperson
objectclass = eperson
objectclass = top
objectclass = person
```

where

```
ou=MyUnit,o=MyOrg
ou=MyUnit
objectclass=top
objectclass=organizationalUnit
description=Tivoli organization
```

While defining the LDAP repository, `Object Classes` and `Search bases` have been adapted to this LDAP schema.

Using the Pluggable Authentication Module

IBM Workload Scheduler enhances the WebSphere Application Server by supporting a user authentication mechanism based on the Pluggable Authentication Module.

This enhancement provides a single authentication mechanism that is capable of authenticating users no matter what their user registry implementations are based on, local OS or LDAP.

IBM Workload Scheduler automatically installs the plug-in that enables WebSphere Application Server to use enabled Pluggable Authentication Module authentication. The plug-in uses the service with name `other`. Ordinarily, you need do nothing to configure the Pluggable Authentication Module. However, if the level of your authorizations inhibits you from using `other`, you should add the service with name `checkpassword` in the `/etc/pam.conf` file.

The use of the Pluggable Authentication Module also extends the capabilities of WebSphere Application Server to include support for authentication in HP Trusted Mode environments.

IBM Workload Scheduler is set by default to use a Pluggable Authentication Module user registry called "custom" or "customPAM". If the Pluggable Authentication Module is not configured with this registry, WebSphere Application Server looks in the local user registry on the master domain manager.

Using the Loadable authentication module

The Loadable authentication module (LAM) performs both authentication and identification on AIX systems.

The Loadable authentication module (LAM) is different from the Pluggable authentication module (PAM), which performs only authentication.

IBM Workload Scheduler automatically installs the plug-in that enables WebSphere Application Server to use PAM-enabled authentication as the default authentication system on UNIX and Linux operating systems.

The LAM is used to provide identification, such as account name and attribute information, and authentication, such as password storage and verification, or both. The AIX security subsystem directs authentication and identification requests to the proper method by using two attributes: **registry** and **SYSTEM**.

Where users and their user attributes are defined (local, LDAP) is reflected by the **registry** user attribute and how users are authenticated (local, NIS, LDAP, Kerberos) is reflected by the **SYSTEM** attribute.

IBM Workload Scheduler uses a custom registry module to integrate LAM in WebSphere Application Server. You can activate LAM on AIX systems by setting the **activeUserRegistry** property to `CustomLAM`, and then running the changeSecurityProperties.sh script indicating the property file to update the value.

Chapter 6. Network administration

This chapter describes how to administer the IBM Workload Scheduler network. It has the following topics:

Network overview

A IBM Workload Scheduler network consists of one or more domains arranged hierarchically. A IBM Workload Scheduler domain is a logical grouping of workstations, consisting of a domain manager and a number of agents.

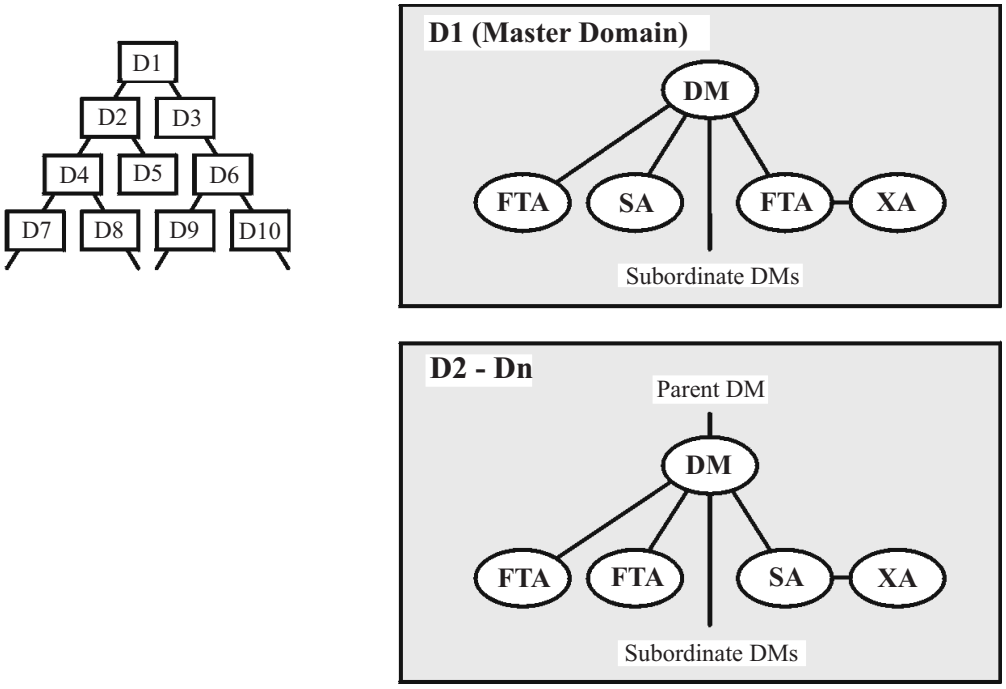


Figure 2. IBM Workload Scheduler network domain structure

Network definition

Domain
A named group of IBM Workload Scheduler workstations consisting of one or more agents and a domain manager. All domains have a parent, except the master domain.

Master domain
The topmost domain in an IBM Workload Scheduler network.

Master domain manager
The domain manager in the topmost domain of an IBM Workload Scheduler network. It contains the centralized master files used to document scheduling objects. It creates the Production Control file (Symphony) at the start of each production period and performs all logging and reporting for the network. See also Domain Manager.

Backup master domain manager
A fault-tolerant agent capable of assuming the responsibilities of the master domain manager.

Parent domain
The domain directly above the current domain. All domains, except the master domain, have a parent domain. All communications to/from a domain is rooted through the parent domain manager.

Domain Manager
The management hub in a domain. All communications in and from the agents in a domain is routed through the domain manager. See also Master Domain Manager.

Backup domain manager
A fault-tolerant agent capable of assuming the responsibilities of its domain manager.

Fault-tolerant agent
An agent workstation capable of resolving local dependencies and launching its jobs in the absence of a domain manager.

Standard agent
An agent workstation that launches jobs only under the direction of its domain manager.

Extended agent
An agent workstation that launches jobs only under the direction of its host. Extended agents can be used to interface IBM Workload Scheduler with non-IBM Workload Scheduler systems and applications

Dynamic agent
A workstation that manages a wide variety of job types, for example, specific database or FTP jobs, in addition to existing job types. This workstation is automatically created and registered when you install the dynamic agent. Because the installation and registration processes are performed automatically, when you view the agent in the Dynamic Workload Console, it results as updated by the Resource Advisor Agent. You can group agents in pools and dynamic pools.

In a simple configuration, dynamic agents connect directly to a master domain manager or to a dynamic domain manager. However, in more complex network topologies, if the network configuration prevents the master domain manager or the dynamic domain manager from directly

communicating with the dynamic agent, then you can configure your dynamic agents to use a local or remote gateway.

Host The scheduling function required by extended agents. It can be performed by any IBM Workload Scheduler workstation, except another extended agent.

Network communications

In a IBM Workload Scheduler network, agents communicate with their domain managers, and domain managers communicate with their parent domain managers. There are basically two types of communications that take place:

- Start-of-production period initialization (distribution of new Symphony file)
- Scheduling events in the form of change-of-state messages during the production period

Before the start of each new production period, the master domain manager creates a production control file called Symphony. Then, IBM Workload Scheduler is restarted in the network, and the master domain manager sends a copy of the new `Symphony` file to each of its automatically-linked agents and subordinate domain managers. The domain managers, in turn, send copies to their automatically-linked agents and subordinate domain managers. Agents and domain managers that are not set up to link automatically are initialized with a copy of `Symphony` as soon as a link operation is run in IBM Workload Scheduler.

Once the network is started, scheduling messages, like job starts and completions, are passed from the agents to their domain managers, through parent domain managers to the master domain manager. The master domain manager then broadcasts the messages throughout the hierarchical tree to update the `Symphony` files of all domain managers and the domain managers forward the messages to all fault-tolerant agents in their domain running in *FullStatus* mode.

Network links

Links provide communications between IBM Workload Scheduler workstations in a network. Links are controlled by the AUTO Link flag, and the Console Manager **link** and **unlink** commands. When a link is open, messages are passed between two workstations. When a link is closed, the sending workstation stores messages in a local `pobox` file and sends them to the destination workstation when the link is reopened.

This means that when links are closed, the message queues fill up with messages for the inaccessible workstations. To maximize the performance of IBM Workload Scheduler, monitor workstations for closed links and attempt to reopen them as soon as possible.

Note: Extended agents do not have links. They communicate with their domain managers through their hosts.

To have a workstation link opened automatically, turn on the AUTO Link flag in the workstation's definition. The link is first opened when IBM Workload Scheduler is started on the Master Domain workstation. If the subdomain manager and workstations are not initialized and their AUTO Link flag is on, the master domain manager attempts to link to its subordinates and begin the initialization processes. If the AUTO Link flag is turned off, the workstation is only initialized

by running a **link** command from the master domain manager. After the workstation is initialized, it automatically starts and issues a link back to its domain manager.

If you stop a workstation, the links from it to other workstations are closed. However, the links from the other workstations to it remain open until either one of the following situations occurs:

- The stopped workstation is restarted and a **link** command is issued
- The other workstations' **mailman** processes time out, and perform an **unlink** for the workstation

When the **link** command is issued and the connection has been established, if the domain manager does not receive any reply within the timeout period, the `chkhltst` service is automatically invoked by **mailman**.

This service verifies that the workstation mailbox can be successfully read, and checks if there are errors in the mailbox header. Resulting information is logged in the `TWSMERGE.log` file of the domain manager as follows:

- If a file system error occurs while opening the mailbox, the following message is reported: `AWSBDY126E An error occurred opening the Mailbox.msg file in CPU_NAME.`
- If an error occurs while opening the mailbox because **mailman** is reading the mailbox, the following message is reported: `AWSBDY123I The Mailbox.msg file in CPU_NAME is correctly read by Mailman.`
- If the mailbox is correctly opened, but an error occurs while reading the header, the following message is reported: `AWSBDY125E An error occurred reading the header of the Mailbox.msg file in CPU_NAME.`
- If the mailbox is correctly opened and no error occurs while reading the header, the following message is reported: `AWSBDY124W The Mailbox.msg file in CPU_NAME is not read by Mailman.`

This service can also be launched manually by using the **conman** command. See the *IBM Workload Scheduler User's Guide and Reference* for more details.

To be certain that inter-workstation communication is correctly restored, you can issue a **link** command after restarting a workstation.

Working across firewalls

In the design phase of a IBM Workload Scheduler network, the administrator must know where the firewalls are positioned in the network, which fault-tolerant agents and which domain managers belong to a particular firewall, and which are the entry points into the firewalls. When this has been clearly understood, the administrator should define the **behindfirewall** attribute for some of the workstation definitions in the IBM Workload Scheduler database. In particular, if a workstation definition is set with the **behindfirewall** attribute to ON, this means that there is a firewall between that workstation and the IBM Workload Scheduler master domain manager. In this case, the workstation-domain manager link is the only link allowed between the workstation and its domain manager.

All IBM Workload Scheduler workstations should be defined with the **behindfirewall** attribute if the link with the corresponding domain manager, or with any domain manager in the IBM Workload Scheduler hierarchy right up to the master domain manager, is across a firewall.

When mapping an IBM Workload Scheduler network over an existing firewall structure, it does not matter which fault-tolerant agents and which domain managers are on the secure side of the firewall and which ones are on the non secure side. Firewall boundaries should be the only concern. For example, if the master domain manager is in a non secure zone and some of the domain managers are in secured zones, or vice versa, does not make any difference. The firewall structure must always be considered starting from the master domain manager and following the IBM Workload Scheduler hierarchy, marking all the workstations that have a firewall between them and their corresponding domain manager.

For all workstations with **behindfirewall** set to ON, the conman **start** and **stop** commands on the workstation, and the **showjobs** commands are sent following the domain hierarchy, instead of making the master domain manager or the domain manager open a direct connection to the workstation. This makes a significant improvement in security.

This attribute works for multiple nested firewalls as well. For extended agents, you can specify that an extended agent workstation is behind a firewall by setting the **behindfirewall** attribute to ON, on the host workstation. The attribute is read-only in the plan; to change it in the plan, the administrator must update it in the database and then re-create the plan.

See the *IBM Workload Scheduler: User's Guide and Reference* for details on how to set this attribute.

Configuring dynamic agent communications through a gateway

In some complex network topologies, the master domain manager or the dynamic domain manager are prevented from directly communicating with the dynamic agent.

Before you begin

In a simple configuration, dynamic agents connect directly to the master domain manager or to the dynamic domain manager. However, in more complex network topologies, if the network configuration prevents the master domain manager or the dynamic domain manager from directly communicating with the dynamic agent, for example, if the agents are behind a firewall and need to communicate through the internet, or if they need to communicate with a Network Address Translation (NAT) process, then you can configure your dynamic agents to use a local or remote gateway.

About this task

You can set up your dynamic agents to use a gateway for communication with the master domain manager or to the dynamic domain manager when you install a dynamic agent, or you can configure a gateway subsequent to the installation.

For information about the gateway parameters available with the installation of a dynamic agent, see the section about agent installation parameters in *Planning and Installation Guide*.

To configure an existing IBM Workload Scheduler version 9.2 or later dynamic agent to communicate to its master domain manager or dynamic domain manager through a local gateway, perform the following configuration steps:

Procedure

1. Edit the `JobManager.ini` file on the dynamic agent workstation that you want to configure to communicate through a gateway. Edit the [ResourceAdvisorAgent] section so that the value of the **`ResourceAdvisorURL`** parameter is `https://$(tdwb_server):$(tdwb_port)/ita/JobManagerGW/JobManagerRESTWeb/JobScheduler/resource` , where, $(*tdwb_server*) and $(*tdwb_port*) correspond to the host name and port of the gateway that you want to use for communication with the master domain manager or the dynamic domain manager.
2. Stop and start the dynamic agent to implement the changes.

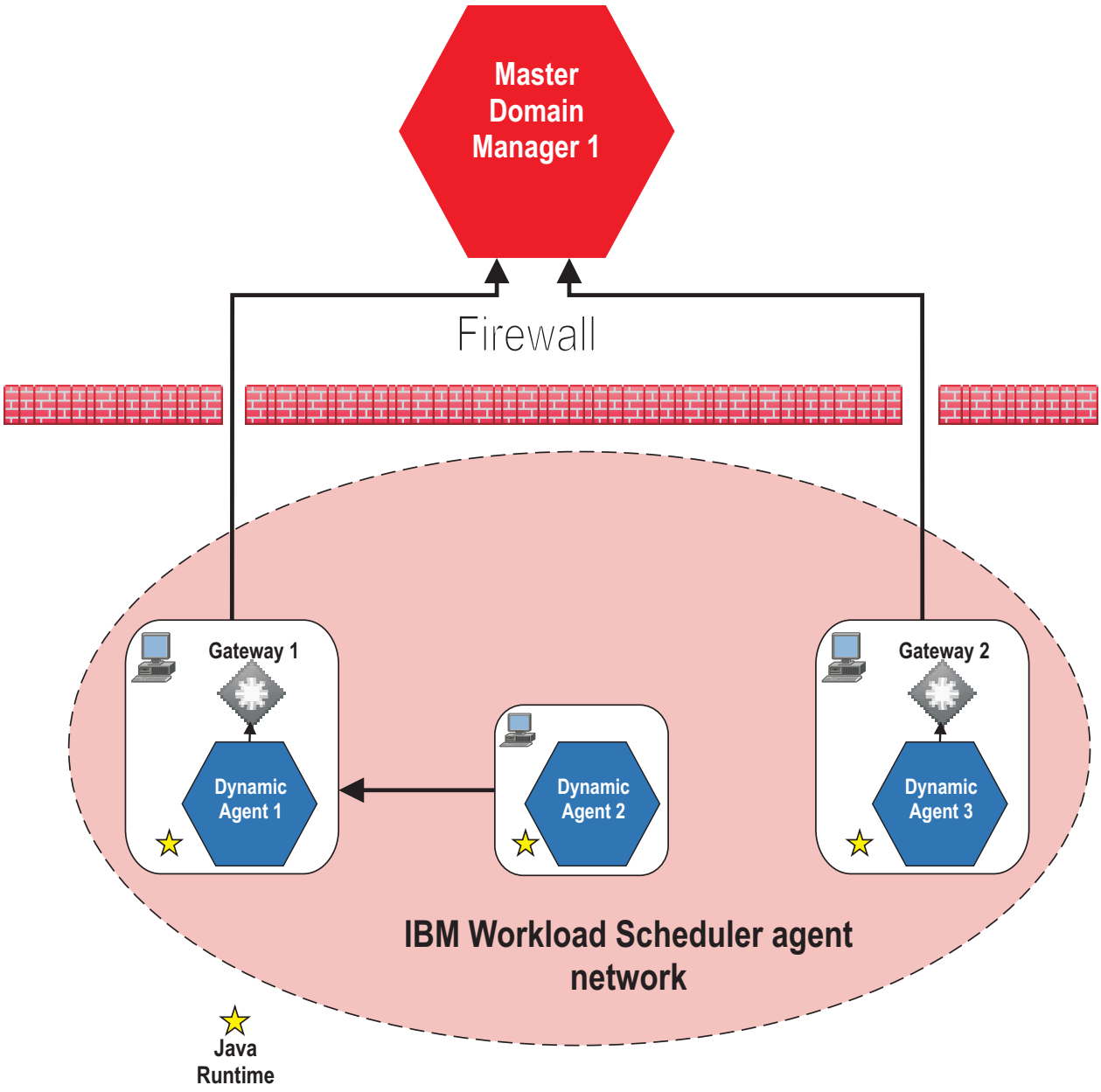
Results

The master domain manager or dynamic domain manager can now communicate with the dynamic agent workstation through the gateway.

Note: If you have more than 100 dynamic agents that communicate through one single gateway to the master domain manager or dynamic domain manager, in the `JobManagerGW.ini` file on the dynamic agent workstation where the gateway resides, set the `ActionPollers` parameter as described in "Configuring general properties [ITA]" on page 58.

Example

The following diagram depicts a network topology where the master domain manager communicates to the dynamic agents, located behind a firewall, through a gateway configured on one of the dynamic agents.



The following are the configuration settings used in the network topology depicted in the figure:

Table 61. Configuration settings

Dynamic Agent	Configuration File	Parameter	Value

Dynamic Agent 1 - Local gateway	JobManager.ini	Section
		`[ResourceAdvisorAgent]`
		`ResourceAdvisorUrl`
			`https://`
			`$(`*`tdwb_server`*`):`
			`$(`*`tdwb_port`*`)/ita/`
			`JobManagerGW/`
			`JobManagerRESTWeb/`
			`JobScheduler/resource`
			
			where,
			$(*tdwb_server***)**
			The host name of the Dynamic Agent 1 workstation.
			$(*tdwb_port***)**
			The port number of the Dynamic Agent 1 workstation.

Dynamic Agent 2 - Remote gateway	JobManager.ini	Section
		`[ResourceAdvisorAgent]`
		`ResourceAdvisorUrl`
			`https://`
			`$(`*`tdwb_server`*`):`
			`$(`*`tdwb_port`*`)/ita/`
			`JobManagerGW/`
			`JobManagerRESTWeb/`
			`JobScheduler/resource`
			
			where,
			$(*tdwb_server***)**
			The host name of the Dynamic Agent 1 workstation.
			$(*tdwb_port***)**
			The port number of the Dynamic Agent 1 workstation.

Dynamic Agent 3 - Local gateway	JobManager.ini	Section
		`[ResourceAdvisorAgent]`
		`ResourceAdvisorUrl`
			`https://`
			`$(`*`tdwb_server`*`):`
			`$(`*`tdwb_port`*`)/ita/`
			`JobManagerGW/`
			`JobManagerRESTWeb/`
			`JobScheduler/resource`
			
			where,
			$(*tdwb_server***)**
			The host name of the Dynamic Agent 3 workstation.
			$(*tdwb_port***)**
			The port number of the Dynamic Agent 3 workstation.

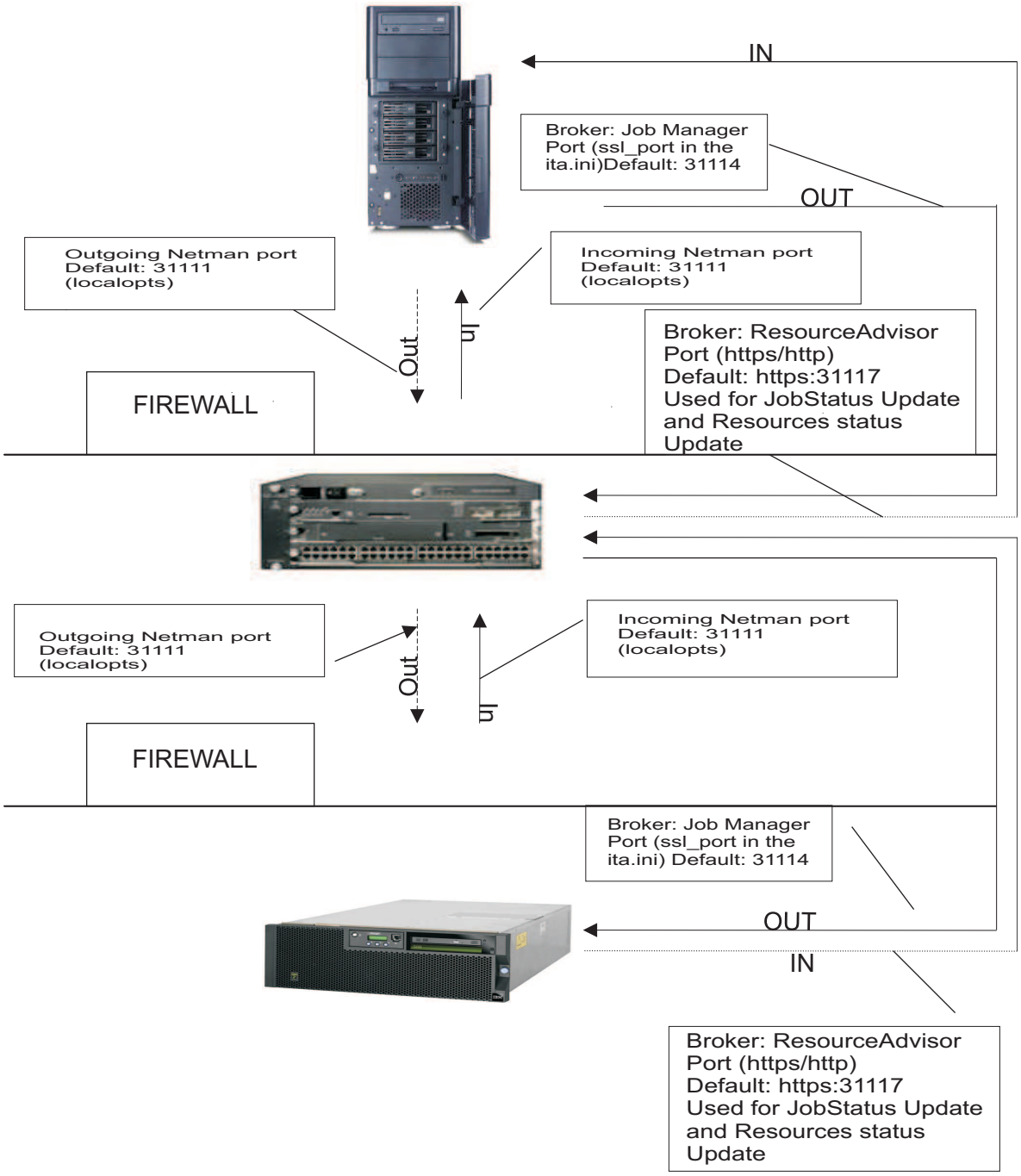
What to do next

For more information about the parameters in the `JobManager.ini` and `JobManagerGW.ini` files, see "Configuring the agent" on page 56.

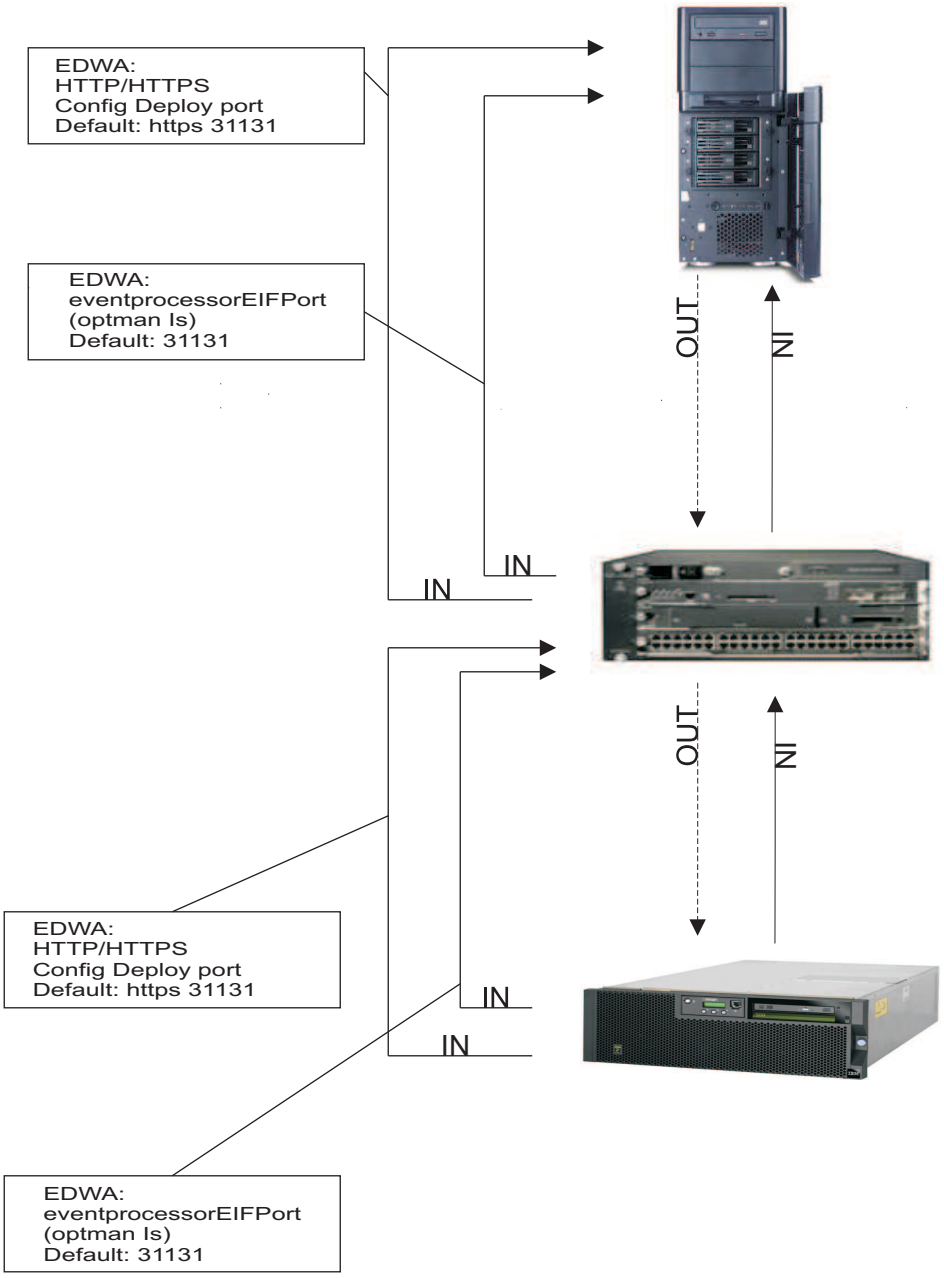
To see an example of the installation parameters that must be specified to configure a gateway when installing a dynamic agent, see the section containing example dynamic agent gateway installations in the *Planning and Installation Guide*.

Enabling Ports

When you install the master domain manager in a IBM Workload Scheduler network all the incoming and outgoing ports are shown in the figure below:



If you enable the event driven workload automation (EDWA) behind the firewall feature the figure below shows all the incoming and outgoing ports.



Network operation

The batchman process on each domain manager and fault-tolerant agent workstation operates autonomously, scanning its `Symphony` file to resolve dependencies and launch jobs. Batchman launches jobs via the jobman process. On a standard agent, the jobman process responds to launch requests from the domain manager's batchman.

The master domain manager is continuously informed of job launches and completions and is responsible for broadcasting the information to domain managers and fault-tolerant agents so they can resolve any inter-workstation dependencies.

The degree of synchronization among the `Symphony` files depends on the setting of the *FullStatus* mode in a workstation's definition. Assuming that these modes are turned on, a fault-tolerant agent's Symphony file contains the same information as the master domain manager's (see the section that explains how to manage workstations in the database in the *IBM Workload Scheduler: User's Guide and Reference*).

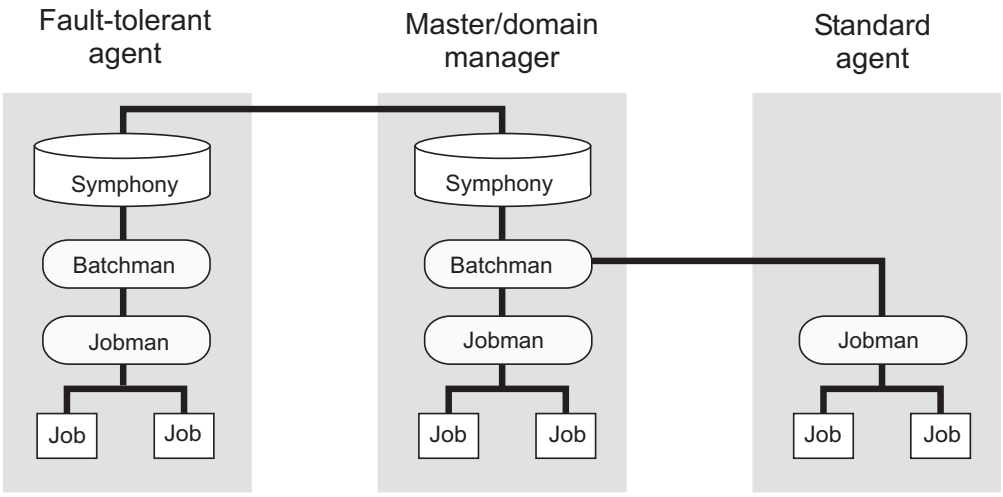


Figure 3. Symphony file synchronization

Network processes

Netman is started by the **`StartUp`** script (command). The order of process creation is netman, mailman, batchman, and jobman. On standard agent workstations, batchman does not run. All processes, except jobman, run as the **TWS** user. Jobman runs as **root**.

When network activity begins, netman receives requests from remote mailman processes. Upon receiving a request, netman creates a writer process and passes the connection off to it. Writer receives the message and passes it to the local mailman. The writer processes (there might be more than one on a domain manager) are started by link requests and are stopped by unlink requests (or when the communicating mailman terminates).

Domain managers, including the master domain manager, can communicate with a large number of agents and subordinate domain managers. For improved efficiency, you can define mailman servers on a domain manager to distribute the communications load (see the section that explains how to manage workstations in the database in the *IBM Workload Scheduler: User's Guide and Reference*).

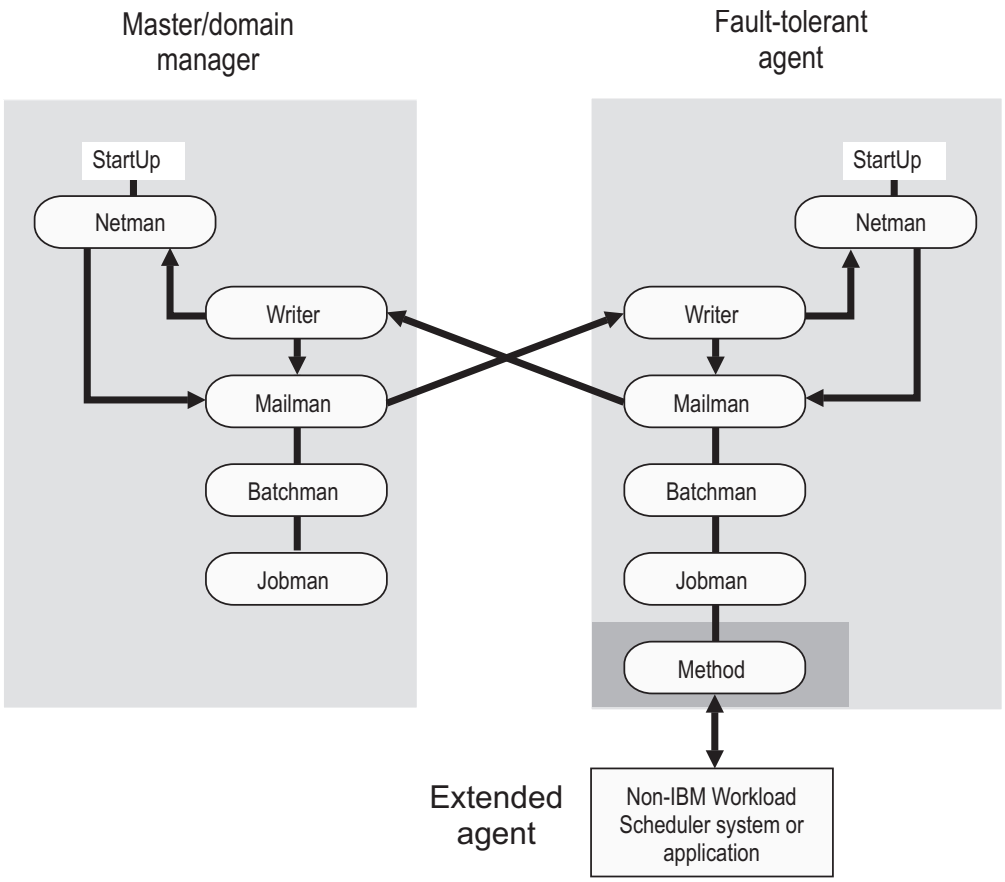


Figure 4. Process creation on domain manager and fault-tolerant agent

The **StartUp** command is normally run automatically, but can also be run manually, as follows:

StartUp

Starts **netman**, the IBM Workload Scheduler network management process.

In Windows, the **netman** service is started automatically when a computer is restarted. **StartUp** can be used to restart the service if it is stopped for any reason.

In UNIX, the **StartUp** command can be run automatically by invoking it from the `/etc/inittab` file, so that WebSphere Application Server infrastructure and **netman** is started each time a computer is rebooted. **StartUp** can be used to restart **netman** if it is stopped for any reason.

The remainder of the process tree can be restarted with the

```
conman start
conman startmon
```

commands. See the documentation about **conman** in the *IBM Workload Scheduler: User's Guide and Reference* for more information.

Note: If you start the StartUp command using a remote shell, the netman process maintains the shell open without returning the prompt. To avoid this problem, modify the StartUp command so that the netman process is called in the background, as follows:

```
# Start netman
/usr/local/TWS851/mae851/TWS/bin/netman&
```

Authorization

You must have ***start*** access to the workstation.

Syntax

StartUp [–v | –u]

Arguments

–v Displays the command version and exits.

–u Displays command usage information and exits.

Examples

To display the command name and version, run the following command:

```
StartUp -v
```

To start the **netman** process, run the following command:

```
StartUp
```

Monitoring the IBM Workload Scheduler processes

You can use event-driven workload automation (EDWA) to monitor the status of network processes and to start a predefined set of actions when one or more specific events take place. For more information about event-driven workload automation, refer to *IBM Workload Scheduler: User's Guide and Reference*.

You can monitor the following processes:

- agent
- appservman
- batchman
- jobman
- mailman
- monman
- netman

The .XML file contains the definition of a sample event rule to monitor the status of the specified processes on the specified workstation. This event rule calls the MessageLogger action provider to write a message in a log file in an internal auditing database. If the condition described in the rule is already existing when you deploy the rule, the related event is not generated. For more information about the MessageLogger action provider, refer to IBM Workload Scheduler User's Guide and Reference:

```
<eventRule name="PROCESSES" ruleType="filter" isDraft="no">
 <eventCondition name="twsProcMonEvt1" eventProvider="TWSApplicationMonitor"
 eventType="TWSProcessMonitor">
   <scope>
    AGENT, BATCHMAN DOWN
   </scope>
   <filteringPredicate>
    <attributeFilter name="ProcessName" operator="eq">
     <value>process_name1</value>
    </attributeFilter>
```

```
    <attributeFilter name="TWSPath" operator="eq">
     <value>TWS_path</value>
    </attributeFilter>
    <attributeFilter name="Workstation" operator="eq">
     <value>workstation_name</value>
    </attributeFilter>
    <attributeFilter name="SampleInterval" operator="eq">
     <value>sample_interval</value>
    </attributeFilter>

   </filteringPredicate>
  </eventCondition>
  <action actionProvider="MessageLogger" actionType="MSGLOG" responseType="onDetection">
   <scope>
    OBJECT=AAAAAAA MESSAGE=TWS PROCESS DOWN: %{TWSPROCMONEVT1.PROCESSNAME}
ON %{TWSPROCMONEVT1.TWSPATH}
   </scope>
   <parameter name="ObjectKey">
    <value>object_key</value>
   </parameter>
   <parameter name="Severity">
    <value>message_severity</value>
   </parameter>
   <parameter name="Message">
    <value>log_message</value>
   </parameter>
  </action>
 </eventRule>
</eventRuleSet>
```

where:

process_name
: Is the name of the process to be monitored. You can insert more that one process name, as follows:

```
<attributeFilter name="ProcessName" operator="eq">
     <value>agent</value>
     <value>batchman</value>
    </attributeFilter>
```

TWS_path
: Is the directory containing the Symphony file and the `bin` directory.

workstation_name
: Is the workstation on which the event is generated.

sample_interval
: Is the interval, expressed in seconds, for monitoring the process status.

object_key
: Is a key identifying the object to which the message pertains.

message_severity
: Is the severity of the message.

log_message
: Is the message to be logged.

Optimizing the network

The structure of a IBM Workload Scheduler network goes hand in hand with the structure of your enterprise's network. The structure of the domains must reflect the topology of the network in order to best use the available communication channels.

But when planning the IBM Workload Scheduler network, the following must be taken into consideration:

- Data volumes
- Connectivity

Data volumes

Network capacity must be planned to adapt to the amount of data that is circulating. Particularly high transmission volumes might be caused by the following:

- Transfer of large Symphony files.
- Message traffic between the master domain manager and a *FullStatus* agent.
- Message traffic from a domain manager when the domain has many agents.
- Heavy use of internetwork dependencies, which extends traffic to the entire network.

Connectivity

For the more critical agents in your network, you need to consider their position in the network. The reliability of workload execution on a particular agent depends on its capacity to receive a fresh Symphony file at the start of the production period. If the workload contains many dependencies, a reliable connection to the rest of the network is also required. These factors suggest that the best place for critical agents is in the master domain, or to be set up as domain managers immediately under the master domain manager, possibly receiving their Symphony files through a set of dedicated mailman servers. Further, it is important for critical agents that any domain manager above them in the tree structure must be hosted on powerful systems and must have an adequate backup system to ensure continuity of operation in the event of problems.

IBM Workload Scheduler provides two mechanisms to accommodate a particular network situation: the domain structure and mailman servers. Whereas domain structure establishes a hierarchy among IBM Workload Scheduler agents, mailman servers are used to tune the resources dedicated to the connection between two agents.

Domain

Use the IBM Workload Scheduler domain structure mechanism to create a tree-shaped structure for the network, where all communications between two points use the unique path defined by the tree (climb to the common ancestor and go down to the target, as opposed to direct TCP communication). As a consequence, the domain structure separates the network into more-manageable pieces. This is for easier filtering, overview, action, and monitoring. However, it does also introduce some delay in the workload processing. For instance when distributing the Symphony file, a fault-tolerant agent inside a domain needs to wait for two steps of Symphony distribution to be completed (from master domain manager to domain manager and from domain manager to fault-tolerant agent). The same is valid for every other type of communication that comes from the master domain manager.

This has the following implications:

- Critical business activities must be as close as possible to the master domain manager
- The domain manager must be installed on as powerful a workstation as possible

- A similarly powerful backup domain manager must be included in the network
- The network link between the domain manager and its backup must be as fast as possible to pass all the updates received from the subtree
- If intervention is needed directly on the domain, either give shell access to the operators to use the IBM Workload Scheduler command line, or install a connector so that the Dynamic Workload Console can be used.

Mailman servers

Mailman servers allocate separate processes dedicated to the communication with other workstations. The main mailman is dedicated to the transfer and network hub activities. The use of mailman servers on the domain manager must be carefully planned. The main parameter is the number of downstream connections at each level of the tree. This number describes the number of mailman servers that a main mailman is connected to, or the number of agents a mailman server is connected to. The maximum number of downstream connections is about 20 for Solaris, 50 for Windows and about 100 for other UNIX workstations, depending on their power. Typical downstream connections is about 10 for Solaris, about 15 for Windows and about 20 for other UNIX workstations. However, you must also take into consideration the link speed and the queue sizes, discussed below.

Planning space for queues

In order to plan space for event queues, and possible alert levels and reactions, it is necessary to model the flows passing through the agents, and the domain managers in particular.

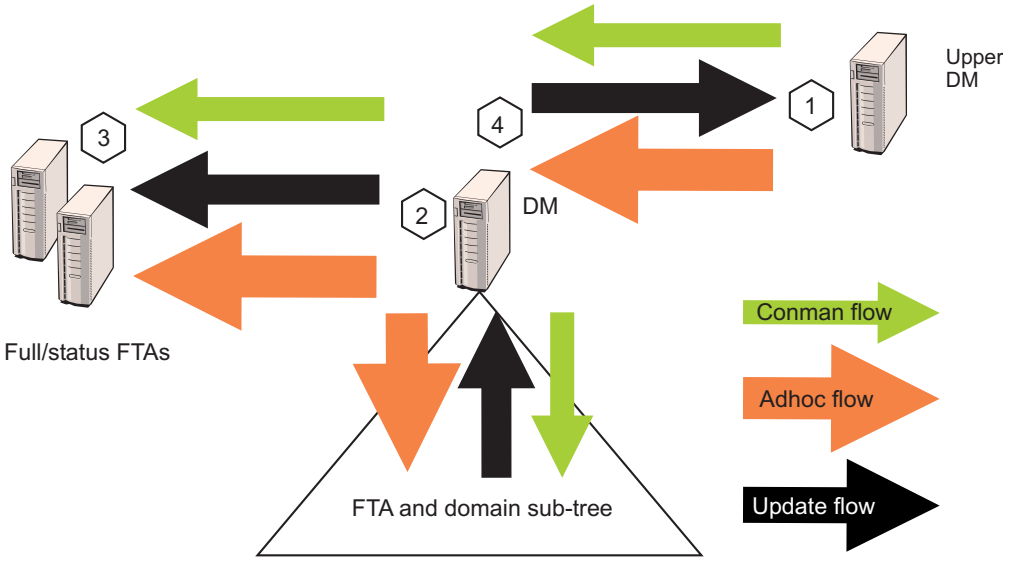


Figure 5. Typical IBM Workload Scheduler network flows.

For a typical domain manager, the main flow comes from update activity reported by the sub tree, and from ad hoc submissions arriving from the master domain manager and propagating to the entire network. Under these conditions, the most critical errors are listed by order of importance in Table 62 on page 288:

Table 62. Critical flow errors

Flow no.	Location	Queue	Risk	Impact
1	Upper domain manager	dm.msg	The queue fills up because of too many unlinked workstations in the domain or a downstream domain manager has failed.	The upper domain manager fails and propagates the error.
2	Domain manager	*FullStatus* fta.msg	The queue fills because of too many unlinked workstations in the domain or because the *FullStatus* fault-tolerant agent is not coping with the flow.	The domain manager fails and favors the occurrence of #1.
3	Domain manager and *FullStatus* fault-tolerant agent	Mailbox.msg or Intercom.msg	The queue fills because the *FullStatus* fault-tolerant agent cannot cope with flow.	The *FullStatus* fault-tolerant agent fails and favors the occurrence of #2.
4	Domain manager	tomaster.msg	The queue fills because of too many unlinked workstations in the domain.	The domain manager starts to unlink the subtree and accumulates messages in the structure.
5	Fault-tolerant agents - only when `enSwfaultTol` global option is set to *yes*	deadletter.msg	The queue fills because of too many unlinked workstations in the domain.	The agent stops.
6	Fault-tolerant agents - only when `enSwfaultTol` global option is set to *yes*	ftbox.msg	This queue is circular. The rate of messages entering the queue exceeds the rate of messages being processed, because of too many unlinked workstations in the domain.	Events are lost.

Note:

1. Flows are greater at the master domain manager and at any *FullStatus* fault-tolerant agents in the master domain than at subordinate domain managers or *FullStatus* fault-tolerant agents.
2. Use **`evtsize -show`** to monitor queue sizes.
3. The amount of update flow is related to the amount of workload running in a particular subtree and is unavoidable.
4. The amount of ad hoc flow is related to the amount of additional workload on any point of the network. It can be reduced by planning more workload even if it is inactive. Note that simple reruns (not `rerun from`) do not create an ad hoc flow.

The planning, alert, and recovery strategy must take into account the following points:

- Queue files are created with a fixed size and messages are added and removed in a cyclical fashion. A queue reaches capacity when the flow of incoming messages exceeds the outgoing flow for a sufficient length of time to use up the available space. For example, if messages are being added to a queue at a rate of 1MB per time unit and are being processed and removed at a rate of 0.5 MB per time unit, a queue sized at 10 MB (the default) is at capacity after 20 time units. But if the inward flow rate descends to be the same as the outward flow rate after 19 time units, the queue does not reach capacity.

- The risk of the domain manager failing can be mitigated by switching to the backup domain manager. In this case, the contents of the queues on the domain manager are unavailable until the domain manager backup is started. In all cases, the size of the queue on the upper domain manager towards any other domain manager must respect condition A of Table 63.
- The risk that fault-switching fault-tolerant agents might not be able to cope with the flow must be planned beforehand. The specifications for fault-switching fault-tolerant agents must be similar to those of the domain manager, to avoid that an agent receives a load that is not appropriate to its capacity. Check if a queue is forming at the *FullStatus* fault-tolerant agents, both in ordinary and peak operation situations.
- Once risk #2 has been dealt with, the possibility of a network link failure can be mitigated by sizing the queue from a domain manager to the *FullStatus* fault-tolerant agents appropriately as a function of the average network outage duration, and by increasing the size of the mailbox in case of unexpected long outage (see condition B of Table 63).
- The same condition applies for avoiding an overflow of the domain manager's tomaster.msg queue with respect to network outages (see condition C) of Table 63.

Table 63. Queue sizing conditions.

A	MaxAlertTime <= size(UpperDM#queueToDM) / averageAdhocFlow
B	MaxNetOutage <= size(DM#queueToFSFTA) / (averageAdhocFlow + averageUpdateFlow)
C	MaxNetOutage <= size(DM#queueToUpperDM) / (averageUpdateFlow)

Monitoring the IBM Workload Scheduler message queues

You can use event-driven workload automation (EDWA) to monitor the size of message queues and to start a predefined set of actions when one or more specific events take place. For more information about event-driven workload automation, refer to *IBM Workload Scheduler: User's Guide and Reference*.

You can monitor the following message queues:

- appserverbox
- mailbox
- clbox
- intercom
- courier
- monbox
- moncmd
- server
- tomaster
- pobox
- planbox

The following .XML file contains the definition of a sample event rule to monitor the mailbox queue on the specified workstation and send an email when the filling percentage is greater than the specified value. If the condition described in the rule is already existing when you deploy the rule, the related event is not generated. This event rule calls the MailSender action provider to send an email to the

receivers you specify. For more information about the MailSender action provider, refer to *IBM Workload Scheduler: User's Guide and Reference*:

```
<?xml version="1.0"?>
<eventRuleSet  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
  xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules
   http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules/EventRules.xsd">
 <eventRule name="MONITORQUEUE" ruleType="filter" isDraft="no">
  <eventCondition name="twsMesQueEvt1" eventProvider="TWSApplicationMonitor" eventType="TWSMessageQueues">
   <scope>
    MAILBOX FILLED UP 80% ON FTA
   </scope>
   <filteringPredicate>
    <attributeFilter name="MailboxName" operator="eq">
     <value>mailbox_name</value>
    </attributeFilter>
    <attributeFilter name="FillingPercentage" operator="ge">
     <value>filling_percentage</value>
    </attributeFilter>
    <attributeFilter name="Workstation" operator="eq">
     <value>workstation_name</value>
    </attributeFilter>
    <attributeFilter name="SampleInterval" operator="eq">
     <value>sample_interval</value>
    </attributeFilter>
   </filteringPredicate>
  </eventCondition>
  <action actionProvider="MailSender" actionType="SendMail" responseType="onDetection">
   <scope>
    TWSUSER@TWS : THE MAILBOX ON workstation_name...
   </scope>
   <parameter name="To">
    <value>main_receiver_list</value>
   </parameter>
   <parameter name="Subject">
    <value>mail_subject</value>
   </parameter>
  </action>
 </eventRule>
</eventRuleSet>
```

where:

mailbox_name
: Is the name of the mailbox to monitor.

filling_percentage
: Is the filling percentage. Supported operators are as follows:

  ge causes the event generation when the mailbox filling percentage increases over the threshold value. The event is generated only the first time the specified mailbox filling percentage is reached. If you restart the SSM agent and the filling percentage is higher than the threshold value, the event is generated again. Table 64 provides an example in which the **ge** operator is set to 70%.

Table 64. Example for the ge operator

Mailbox name	Filling percentage	Action
Sample (0)	>= 70%	event not generated
Sample (0)	< 70%	event not generated
Sample (n-1)	< 70%	event not generated
Sample (n)	>= 70%	event generated

Table 64. Example for the ge operator (continued)

Mailbox name	Filling percentage	Action
Sample (n+1)	>= 70%	event not generated

le causes the event generation when the mailbox filling percentage decreases under the threshold value. The event is generated only the first time the specified mailbox filling percentage is reached. If you restart the SSM agent and the filling percentage is lower than the threshold value, the event is not generated until the filling percentage increases over the threshold value and then decreases under it again. Table 65 provides an example in which the **le** operator is set to 50%:

Table 65. Example for the le operator

Mailbox name	Filling percentage	Action
Sample (0)	<= 50%	event not generated
Sample (0)	> 50%	event not generated
Sample (n-1)	> 50%	event not generated
Sample (n)	<= 50%	event generated
Sample (n+1)	<= 50%	event not generated

workstation_name
Is the workstation on which the event is generated.

sample_interval
Is the interval, expressed in seconds, for monitoring the mailbox filling percentage.

main_receiver_list
Is the main receiver list.

mail_subject
Is the subject of the mail.

Changing a queue size

Use the **evtsize** command to resize a queue.

When you have used **evtsize** to resize a queue, the queue remain at that size until the next time you use **evtsize**. It only reverts to the default size of 60 MB if you delete it, at which point IBM Workload Scheduler re-creates it with the default size.

evtsize:

Defines the size of the IBM Workload Scheduler message files. This command is used by the IBM Workload Scheduler administrator either to increase the size of a message file after receiving the message, “End of file on events file.”, or to monitor the size of the queue of messages contained in the message file.

Authorization

You must be **maestro** or **root** in UNIX, or **Administrator** in Windows to run **evtsize**. Stop the IBM Workload Scheduler engine before running this command.

Syntax

evtsize -V | -U

evtsize *file_name size*

evtsize -compact *file_name* [*size*]

evtsize -show *file_name*

Arguments

-V Displays the command version and exits.

-U Displays command usage information and exits.

-compact *file_name* **[***size***]**
Reduces the size of the specified message file to the size occupied by the messages present at the time you run the command. You can optionally use this keyword to also specify a new file size.

-show *file_name*
Displays the size of the queue of messages contained in the message file

file_name
The name of the event file. Specify one of the following:

```
Courier.msg
Intercom.msg
Mailbox.msg
PlanBox.msg
Server.msg
pobox/workstation.msg
```

size The maximum size of the event file in bytes. When first built by IBM Workload Scheduler, the maximum size is set to 10 MB.

Note: The size of the message file is equal to or bigger than the real size of the queue of messages it contains and it progressively increases until the queue of messages becomes empty; as this occurs the message file is emptied.

Examples

To set the maximum size of the `Intercom.msg` file to 20 MB, run the following command:

```
evtsize Intercom.msg 20000000
```

To set the maximum size of the `pobox` file for workstation `chicago` to 15 MB, run the following command:

```
evtsize pobox\chicago.msg 15000000
```

The following command:

```
evtsize -show Intercom.msg
```

returns the following output:

```
IBM Workload Scheduler (UNIX)/EVTSIZE 9.4 (1.2.2.4) Licensed Materials -
Licensed Materials - Property of IBM* and HCL**
5698-WSH
```

```
(C) Copyright IBM Corp. 1998, 2016 All rights reserved.
(C) Copyright HCL Technologies Ltd. 2016 All rights reserved
* Trademark of International Business Machines
** Trademark of HCL Technologies Limited
AWSDEK703I Queue size current 240, maximum 10000000 bytes (read 48, write 288)
```

where:

880 Is the size of the current queue of the `Intercom.msg` file

10000000
Is the maximum size of the `Intercom.msg` file

read 48
Is the pointer position to read records

write 928
Is the pointer position to write records

Tuning mailman servers

Once the distribution of agents to mailman servers has been established, all the groups of agents attached to the same server must respect the link condition.

The link condition relates the number of agents connected to a mailman process and the tuning parameters for unlink on the mailman and writer side.

No_agents(i)
The number of agents connected to a given mailman server *i*

Mm_unlink
A parameter set in the `localopts` of both domain manager and agent. Specifies the maximum number of seconds mailman waits before unlinking from a workstation that is not responding.

Wr_unlink
A parameter set in the `localopts` of both domain manager and agent. Specifies the number of seconds the writer process waits before exiting if no incoming messages are received.

Max_down_agents
The maximum probable number of agents that are unavailable without having the `ignore` flag set in the database and having the `autolink` flag on.

tcp timeout
A parameter set in the `localopts` of both domain manager and agent. Specify the maximum number of seconds that can be waited for the completion of a TCP/IP request on a connected workstation that is not responding.

The condition is:

```
Wr_unlink = Mm_unlink > 1.2 * Max_down_agents * tcp timeout
```

This condition expresses that if the time before unlink is smaller than the probable time of idle waiting of the mailman process (waiting connect timeout for each agent that is currently down) in its loop to reactivate the connections, the agents unlink constantly when some agents are down.

Netman configuration file

The netman configuration file exists on all IBM Workload Scheduler workstations to define the services provided by netman. It is called *<TWA_home>*/TWS/network/Netconf. The NetConf file includes comments describing each service. The services are:

2001 Start a writer process to handle incoming messages from a remote mailman.

2002 Start the mailman process. Mailman, in turn, starts the rest of the process tree (batchman, jobman).

2003 Stop the IBM Workload Scheduler process to handle incoming messages from a remote mailman.

2004 Find and return a stdlist file to the requesting Conman process.

2005 Switch the domain manager in a domain.

2006 Locally download scripts scheduled by an IBM Workload Scheduler for z/OS master domain manager.

2007 Required to bypass a firewall.

2008 Stop IBM Workload Scheduler workstations in a hierarchical fashion

2009 Runs the switchmgr script to stop and restart a manager in such a way that it does not open any links to other workstations until it receives the *switchmgr* event. Can only be used when the enSwfaultTol global option is set to *yes*.

2010 Starts mailman with the parameter *demgr*. It is used by the service *2009*. Can only be used when the enSwfaultTol global option is set to *yes*.

2011 Runs **monman** as a child process (son bin/monman.exe)

2012 Runs **conman** to stop the event monitoring engine (command bin/conman.exe stopmon).

2013 Runs **conman** to switch event processors (command bin/conman.exe switchevtproc -this)

2014 Runs **conman** to start event processing (command bin/conman.exe startevtproc -this)

2015 Runs **conman** to stop event processing (command bin/conman.exe stopevtproc -this)

2016 Runs **conman** to force the update of the monitoring configuration file for the event monitoring engine (command bin/conman.exe deployconf)

2017 Runs **conman** to stop event processing on a client (client bin/conman.exe synchronizedcmd -stopevtproc)

2018 Runs **conman** to check event processing on a client (client bin/conman.exe synchronizedcmd -checkevtproc)

2021 Runs **conman** to start appservman

2022 Runs **conman** to run the subcommand **stopappserver** that stops the application server

2023 Runs **conman** to run the subcommand **startappserver** that starts the application server

2501 Check the status of a remote job.

2502 Start the Console Manager – a service requested by the client side of the Remote Console. See the *IBM Tivoli Remote Control: User's Guide* for more information.

2503 Used by the connector to interact with r3batch extended agent.

Determining internal Symphony table size

The mailman service (2002) can optionally take a parameter that determines the initial size of the internal Symphony table. If you do not supply this parameter, mailman calculates the initial table size based on the number of records in the file.

Note: Mailman expands the table if it needs to, even if this parameter is not supplied.

In normal circumstances, leave mailman to take the default value in the `NetConf` file as supplied (32000). However, if you are experiencing problems with memory, you can allocate a table that is initially smaller. To do this you change the parameter to the service *2002* in the `NetConf` file. The syntax for the entry is:

```
2002     son     bin/mailman [ -parm <number> ]
```

where, <*number*> is used to calculate the initial Symphony table size based on the number of records in the Symphony file.

If *r* is the number of records in the Symphony file when batchman starts, Table 66 shows how the size of the internal Symphony table is calculated, depending on the value of <*number*>:

Table 66. Calculation of internal Symphony table

Value of <*number*>	**Table size**
0	(4/3r) + 512
n	if n > r, n if n <= r, (4/3r) + 512
-1	65535
-n	if +n => r, n if +n < r, r + 512

If during the production period you add more jobs, the maximum internal Symphony table size is increased dynamically, up to the maximum number of records allowed in the Symphony file, which is 2,000,000,000.

Defining access methods for agents

Access methods are used to extend the job scheduling functions of IBM Workload Scheduler to other systems and applications. They run on:

Extended agents
They are logical workstation related to an access method hosted by a physical IBM Workload Scheduler workstation (not another extended agent). More than one extended agent workstation can be hosted by the same IBM Workload Scheduler workstation and use the same access method. The extended agent runs on fault-tolerant agents defined using a

standard IBM Workload Scheduler workstation definition, which gives the extended agent a name and identifies the access method. The access method is a program that is run by the hosting workstation whenever IBM Workload Scheduler submits a job to an external system.

Jobs are defined for an extended agent in the same manner as for other IBM Workload Scheduler workstations, except that job attributes are dictated by the external system or application.

Information about job running execution is sent to IBM Workload Scheduler from an extended agent using the job `stdlist` file. A method options file can specify alternate logins to launch jobs and check *opens* file dependencies. For more information, see the *User's Guide and Reference*.

A physical workstation can host a maximum of 255 extended agents.

dynamic agents and IBM Workload Scheduler for z/OS agents
They communicate with external systems to start the job and return the status of the job. To run access methods on external applications using dynamic agents, you define a job of type **access method**.

Access methods are available on the following systems and applications.
- SAP R/3
- z/OS
- Custom methods
- unixssh
- unixrsh
- Local UNIX (fault-tolerant agents only)

The UNIX access methods included with IBM Workload Scheduler, are described in the related section in *Administration Guide*.

If you are working with dynamic agents, for information about defining IBM Workload Scheduler workstations, see the section that explains how to define workstations in the database in *User's Guide and Reference*. For information about writing access methods, see the section about the access method interface in *User's Guide and Reference*.

More information about access methods is found in *Scheduling Applications with IBM Workload Automation*.

UNIX access methods

IBM Workload Scheduler includes two types of UNIX access methods, local UNIX access methods and remote UNIX access methods.

The Local UNIX access method runs on extended agents. Use the Local UNIX access method to enable a single UNIX workstation to operate as two IBM Workload Scheduler workstations, both of which you can run IBM Workload Scheduler scheduled jobs.

The Remote UNIX access method runs on extended agents and dynamic agents.

On extended agents
Use the Remote UNIX access method to designate a remote UNIX workstation to run IBM Workload Scheduler scheduled jobs without having IBM Workload Scheduler installed on it.

On dynamic agents

Define a job of type **xajob** that runs on dynamic agents. The dynamic agent communicates with the external system to start the job and return the status of the job.

Local UNIX access method running on fault-tolerant agents only

The Local UNIX method can be used to define multiple IBM Workload Scheduler workstations on one workstation: the host workstation and one or more extended agents. When IBM Workload Scheduler sends a job to a local UNIX extended agent, the access method, **unixlocl**, is invoked by the host to run the job. The method starts by running the standard configuration script on the host workstation (*<TWA_home>*/TWS/jobmanrc). If the logon user of the job is permitted to use a local configuration script and the script exists as $HOME/TWS/.jobmanrc, the local configuration script is also run. The job itself is then run either by the standard or the local configuration script. If neither configuration script exists, the method starts the job.

The launching of the configuration scripts, jobmanrc and .jobmanrc is configurable in the method script. The method runs the configuration scripts by default, if they exist. To disable this feature, you must comment out a set of lines in the method script. For more information, examine the script file *<TWA_home>*/TWS/methods/unixlocl on the extended agent's host.

Remote UNIX access method

The Remote UNIX access method can be used to designate a non-IBM Workload Scheduler workstation to run jobs scheduled by IBM Workload Scheduler. You can use unixrsh or unixssh:

The unixrsh access method

When IBM Workload Scheduler sends a job to a remote UNIX extended agent, the access method, unixrsh, creates a /tmp/maestro directory on the non-IBM Workload Scheduler workstation. It then transfers a wrapper script to the directory and runs it. The wrapper then runs the scheduled job. The wrapper is created only once, unless it is deleted, moved, or is outdated.

To run jobs using the unixrsh access method, the job logon users must be given appropriate access on the non-IBM Workload Scheduler UNIX workstation. To give appropriate access, a .rhost, /etc/host.equiv, or equivalent file must be set up on the workstation. On extended agents, if *opens* file dependencies are to be checked, *root* access must also be permitted. Contact your system administrator for help. For more information about the access method, examine the script file *<TWA_home>*/TWS/methods/unixrsh on an extended agent's host.

The unixssh access method

The unixssh access method works like unixrsh but uses a secure remote shell to connect to the remote host. The files used by this method are:

```
methods/unixssh
methods/unixssh.wrp
```

The unixssh method uses the *ssh* key. You can generate this keyword with any tools that are compatible with the secure remote shell.

Note: The passphrase must be blank.
The following scenario gives an example of how to set up the method:

You installed a IBM Workload Scheduler, fault-tolerant agent or dynamic agent with the *TWS_user*: `twsuser`. You want to run a remote shell in the remote host "REMOTE_HOST" with the user "guest". The procedure is as follows:

1. Create the public and private key for the user`twsuser`, The following is an example using rsa:
   a. Log on as `twsuser`
   b. Run
      ```
      ssh-keygen -t rsa
      ```
   c. When the tool asks for the passphrase, press Enter (leaving the passphrase blank.) The keys are saved as follows:

Key	Location	Comment
Public	<*TWA_home*>/TWS/.ssh/id_rsa.pub	
Private	<*TWA_home*>/TWS/.ssh/id_rsa	Do not send this file!

   Note: Different tools store the key in different places.
2. At the remote host, perform the following actions:
   a. Telnet to the remote host.
   b. Log on as "guest".
   c. Change to the `.ssh` directory in the user home directory, or create it if it does not exist (the directory permissions must be adequate: for example, 700 for the directory and 600 for its contents).
   d. Append the *public* key you created in step 1 to the `authorized_keys` file (create the file if it does not exist), using the command:
      ```
      cat id_rsa.pub >> authorized_keys
      ```
3. At the fault-tolerant agent or dynamic agent, make the remote host "known" before attempting to let IBM Workload Scheduler processes use the connection. This action can be achieved in one of two ways:
   - Log on as `twsuser` and connect to the host using the command:
     ```
     ssh -l guest <remote_host_name> ls
     ```
     A prompt is displayed saying that the host is not known, and asking permission to access it. Give permission, and the host is added to the list of known hosts.
   - Alternatively, use the ssh documentation to add the remote host to the file of known hosts.

Managing production for extended agents

In general, jobs that run on extended agents behave like other IBM Workload Scheduler jobs. IBM Workload Scheduler tracks a job's status and records output in the job's `stdlist` files. These files are stored on the extended agent's *host* workstation. For more information on managing jobs, see the section that describes IBM Workload Scheduler plan tasks in the *IBM Workload Scheduler: User's Guide and Reference*.

Failure launching jobs on extended agents and dynamic agents

If the access method is not in the proper directory on the extended agent's host, on the dynamic agent, or the method cannot be accessed by IBM Workload Scheduler, jobs fail to launch or a file dependency is not checked. For a job, the IBM Workload Scheduler jobs logon or the logon specified in the method options file

must have read and execute permissions for the access method. When checking a file to satisfy an *opens* dependency, root is used as the login unless another login is specified in the method options file. For more information about method options, see the *IBM Workload Scheduler: User's Guide and Reference*.

IP address validation

When a TCP/IP connection is established, netman reads the requester's node name and IP address from the socket. The IP address and node name are used to search the `Symphony` file for a known IBM Workload Scheduler workstation with one of the following possible results:

- If an IP address match is found the validation is considered successful.
- If a node name match is found, the validation is considered successful.
- If no match is found in the `Symphony` file or the IP address returned does not match the one read from the socket, the validation is considered unsuccessful.

The local option, `nm ipvalidate`, determines the action to be taken if IP validation is unsuccessful. If the option is set to `full`, unsuccessful validation causes IBM Workload Scheduler to close the connection and generate an error message. If the option is set to `none` (default), IBM Workload Scheduler permits all connections, but generates a warning message for unsuccessful validation checks.

Support for Internet Protocol version 6

IBM Workload Scheduler supports Internet Protocol version 6 (IPv6) in addition to the legacy IPv4. To assist customers in staging the transition from an IPv4 environment to a complete IPv6 environment, IBM Workload Scheduler provides IP dual-stack support. In other terms, the product is designed to communicate using both IPv4 and IPv6 protocols simultaneously with other applications using IPv4 or IPv6.

To this end, the IPv4-specific `gethostbyname` and `gethostbyaddr` functions have been replaced by the new `getaddrinfo` API that makes the client-server mechanism entirely protocol independent.

The `getaddrinfo` function handles both name-to-address and service-to-port translation, and returns `sockaddr` structures instead of a list of addresses These `sockaddr` structures can then be used by the socket functions directly. In this way, `getaddrinfo` hides all the protocol dependencies in the library function, which is where they belong. The application deals only with the socket address structures that are filled in by `getaddrinfo`.

Operating system configuration (UNIX only)

IP validation depends on the system call `getaddrinfo()` to look up all the valid addresses for a host. The behavior of this routine varies, depending on the system configuration. When `getaddrinfo()` uses the file `/etc/hosts`, it returns the first matching entry. If the connection is initiated on an address which appears after the first matching entry, IP validation fails. To resolve the problem, place the entry used to initiate the connection before any other matching entries in the `/etc/hosts` file. If `getaddrinfo()` uses the "named" name server or the Network Information Service server and `getaddrinfo()` fails, contact your system administrator for assistance.

IP address validation messages

Following is a list of the messages for IP validation. If the Local Option `nm ipvalidate` is set to `none` (default), the errors appear as warnings.

See the end of the list of conditions for the key to the variables:

- IBM Workload Scheduler workstation name is not found in the Symphony file

```
Ip address validation failed for request:
Service <num> for <program> on <workstation>(<operating_system_type>).
Connection received from IP address:
<c_ipaddr>. MAESTRO CPU <workstation> not found in
Symphony file.
```

- Call to `getaddrinfo()` fails:

```
IP address validation failed for request:
Service num for <program> on cpu(<operating_system_type>).
Connection received from IP address:
<c_ipaddr>. getaddrinfo() failed, unable to
retrieve IP address of connecting node: <node>.
```

- IP Addresses returned by `getaddrinfo()` do not match the IP address of connection workstation:

```
IP address validation failed for request:
Service <num> for <program> on <workstation>(<operating_system_type>).
Connection received from IP address:
<c_ipaddr>. System known IP addresses for node
name node: <k_ipaddr>.
```

- The IP address specified in the workstation definition for the IBM Workload Scheduler workstation indicated in the service request packet does not match the IP address of connecting workstation:

```
IP address validation failed for request:
Service <num> for <program> on <workstation>(<operating_system_type>).
Connection received from IP address:
<c_ipaddr>. TWS known IP addresses for cpu
<k_ipaddr>.
```

- Regardless of the state of `nm ipvalidate`, the following information message is displayed when IP validation cannot be performed because the `Symphony` file does not exist or an error occurs when reading it:

```
IP address validation not performed for
request: Service <num> for <program> on
<workstation>(<operating_system_type>). Connection received from IP
address: <c_ipaddr>. Cannot open or read
Symphony file. Service request accepted.
```

Where:

<num>
Service number (2001-**writer**, 2002-**mailman**...)

<program>
Program requesting service

<workstation>
IBM Workload Scheduler workstation name of connecting workstation

<operating_system_type>
Operating system of connecting workstation

<node>
Node name or IP address of connecting workstation

<c_ipaddr>
IP address of connecting workstation

<k_ipaddr>
Known IP address for connecting workstation

IP validation is always successful in the absence of a `Symphony` file. In communications from a domain manager to an agent it is normally successful because a `Symphony` file does not yet exist. However, if the agent has a `Symphony` file from a previous run, the initial link request might fail if the `Symphony` file does not include the name of the domain manager.

Impact of network changes

Any changes that you make to your network might have an impact on IBM Workload Scheduler. Workstations can be identified within IBM Workload Scheduler by host name or IP address. Any changes to host names or IP addresses of specific workstations must obviously be also implemented in the IBM Workload Scheduler database. However, remember that if those workstations are involved in jobs that are currently scheduled in the Symphony file, those jobs are looking for the old workstation identity.

Changes to host names or IP addresses of specific workstations can be activated immediately by running **`JnextPlan -for 0000`**. A new production plan is created (containing the updated IP addresses and host names), but the plan time span is not extended.

Thus, plan any network changes with the job schedules in mind, and for major changes you are advised to suspend IBM Workload Scheduler activities until the changes complete in the network and also implemented in the IBM Workload Scheduler database.

Network changes also have a specific impact on the connection parameters used by the application server and the command-line client:

Application server
If you change the network you will need to change the communication parameters specified in the application server configuration files. How to do this is described in the appendix on the utilities supplied with the WebSphere Application Server in the *IBM Workload Scheduler: Planning and Installation Guide*.

Command-line client
When you connect from the command-line client you supply a set of connection parameters. This is done in one of these ways:

From the localopts file
The default method is that the connection parameters in the `localopts` file are customized when the command line client is installed.

From the useropts file

A `useropts` file might have been created for the user in question, containing a version of the connection parameters personalized for the user.

In the command line, individually

When you invoke one of the command-line programs, you can optionally include the parameters as arguments to the command. These override the values in the `localopts` or `useropts` files.

In the command line, in a file

When you invoke one of the command-line programs, you can optionally include the parameters in a file, the name of which is identified as the **`-file`** argument to the command. These override the values in the `localopts` or `useropts` files.

Modify whichever method you are using to incorporate the new network connection details.

Chapter 7. Setting connection security

IBM Workload Scheduler provides default connection security settings when you install. You can perform your connection security customization in your IBM Workload Scheduler environment.

Connection security overview

IBM Workload Scheduler provides a secure, authenticated, and encrypted connection mechanism for communication based on the Secure Sockets Layer (SSL) protocol, which is automatically installed with IBM Workload Scheduler.

IBM Workload Scheduler also provides default certificates to manage the SSL protocol that is based on a private and public key methodology.

If you do not customize SSL communication with your certificates, to communicate in SSL mode, IBM Workload Scheduler uses the default certificates that are stored in the default directories. However, in a production environment, it is recommended that you customize SSL communication with your own certificates as explained in the following scenarios.

You can customize SSL communication with your certificates according to your security requirements.

You can have the following scenarios:

- "Scenario: Connection between the Dynamic Workload Console and the IBM Workload Scheduler component that has a distributed connector" on page 304.
- "Scenario: Connection between dynamic agents and the master domain manager or dynamic domain manager" on page 322.
- "Scenario: SSL Communication across the IBM Workload Scheduler network" on page 326.
- "Scenario: HTTPS for the command-line clients" on page 336.

IBM Workload Scheduler uses the following types of stores:

truststore
: In security, a storage object, either a file or a hardware cryptographic card, where public keys are stored in the form of trusted certificates, for authentication purposes in web transactions. In some applications, these trusted certificates are moved into the application keystore to be stored with the private keys.

keystore
: In security, a file or a hardware cryptographic card where identities and private keys are stored, for authentication and encryption purposes. Some keystores also contain trusted or public keys.

Scenario: Connection between the Dynamic Workload Console and the IBM Workload Scheduler component that has a distributed connector

The Dynamic Workload Console connects in SSL mode with the IBM Workload Scheduler component that has a distributed connector by using the default certificates. You might configure the Dynamic Workload Console to connect in SSL mode by using your certificates.

You can have SSL communication between the Dynamic Workload Console and one of the following components:

- Master domain manager.
- Backup master domain manager.
- Dynamic domain manager.
- Backup dynamic domain manager.
- Agent with distributed connector.

When you are customizing the Dynamic Workload Console settings, make sure the keys have the same password as the keystore where they are saved. The Dynamic Workload Console keystore password must be the same as the Dynamic Workload Console client and IBM Workload Scheduler server.

Note: When you configure the Dynamic Workload Console to connect to different agents with distributed connector, the Dynamic Workload Console truststore must have a certificate for each connector to enable SSL connection.

Overview

For more information about the SSL connection between Dynamic Workload Console and components that has a distributed connector, see "Overview."

SSL connection by using default certificates

For more information about the SSL default connection, see "SSL connection by using the default certificates" on page 306.

SSL connection by using your certificates

For more information about how to create and enable your SSL certificates, see "SSL connection by using your certificates" on page 309.

Overview

Overview of the Dynamic Workload Console SSL connection

To implement the RMI/IIOP over SSL communication between the Dynamic Workload Console and the SOAP internal communication of master domain manager, backup master domain manager, dynamic domain manager, backup dynamic domain manager or agent with distributed connector, you use the server and client security features of WebSphere Application Server.

The SSL security paradigm implemented in the WebSphere Application Server requires two stores to be present on the clients and the server: a keystore containing the private key and a truststore containing the certificates of the trusted counterparts.

Figure 6 shows the server and client keys, and to where they must be exported for the Dynamic Workload Console:

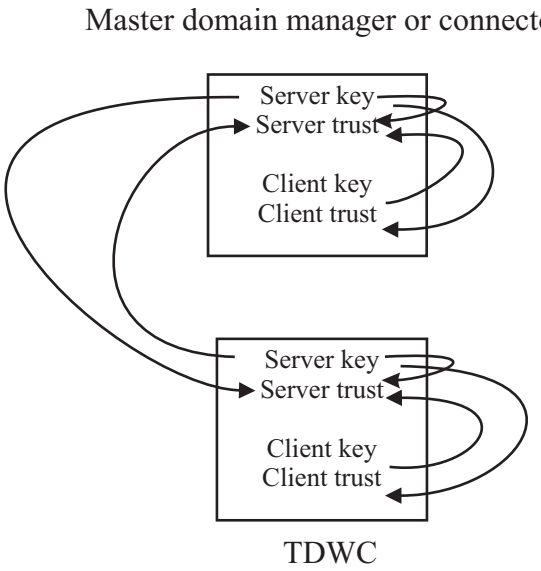


Figure 6. SSL server and client keys

The diagram shows the keys Dynamic Workload Console and components that has distributed connector must extract and distributed to enable SSL communication.The Dynamic Workload Console interface uses the default certificates that are installed in the default keystores to communicate with the agent with distributed connector. You can configure the Dynamic Workload Console to connect in SSL mode with an agent with distributed connector by using your certificates to meet your required security settings.

In addition creating new keys, you can also customize the name, location, and password of the keystore and truststore. For details about possibilities, see Table 67.

Table 67. Changes allowed in IBM Workload Scheduler keystore and truststore

File	Name	Path	Password	New key
TWS server keystore	✔	✔	✔	✔
TWS server truststore	✔	✔	✔	✔
TWS client keystore				✔
TWS client truststore				✔
TDWC client keystore				✔
TDWC client truststore				✔

When you are customizing the Dynamic Workload Console settings, make sure that the keys have the same password as the keystore where they are saved. The Dynamic Workload Console keystore password must be the same as the Dynamic Workload Console client and IBM Workload Scheduler server password.

Note: When you configure the Dynamic Workload Console to connect to different agents with distributed connector, the Dynamic Workload Console truststore must have a certificate for each connector to enable SSL connection.

SSL connection by using the default certificates

The SSL connection between the Dynamic Workload Console and master domain manager, backup master domain manager, dynamic domain manager, backup dynamic domain manager or agent with distributed connector is enabled by using the default certificates.

About this task

You have the following environment:

Dynamic Workload Console installed on the *DWC-WKS* workstation:

- The Dynamic Workload Console is installed in the <DWC_INST_DIR> directory.
- The embedded WebSphere Application Server is installed in the <DWC_INST_DIR>\eWAS directory.

Master domain manager, backup master domain manager, dynamic domain manager, backup dynamic domain manager, or agent with distributed connector installed on the *TWS-WKS* workstation:

- The agent with distributed connector is installed in the <TWS_INST_DIR> directory.
- The embedded WebSphere Application Server is installed in the <TWS_INST_DIR>\eWAS directory.

By default the SSL connection between the Dynamic Workload Console and the component with a distributed connector is enabled by using the default certificates. The default password associated with each of the default keystores is `default`. The SSL connection has the following default certificates:

On Dynamic Workload Console workstations:

Truststore

On Windows operating systems:

- <DWC_INST_DIR>\eWAS\profiles\TIPProfile\etc\TWSServerTrustFile.jks
- <DWC_INST_DIR>\eWAS\profiles\TIPProfile\etc\TWSClientTrustFile.jks

On UNIX operating systems:

- <DWC_INST_DIR>/eWAS/profiles/TIPProfile/etc/TWSServerTrustFile.jks
- <DWC_INST_DIR>/eWAS/profiles/TIPProfile/etc/TWSClientTrustFile.jks

Keystore

On Windows operating systems:

- <DWC_INST_DIR>\eWAS\profiles\TIPProfile\etc\TWSServerKeyFile.jks
- <DWC_INST_DIR>\eWAS\profiles\TIPProfile\etc\TWSClientKeyFile.jks

On UNIX operating systems:

- <DWC_INST_DIR>/eWAS/profiles/TIPProfile/etc/TWSServerKeyFile.jks

- <DWC_INST_DIR>/eWAS/profiles/TIPProfile/etc/TWSClientKeyFile.jks

where *<DWC_INST_DIR>* is the Dynamic Workload Console installation directory.

On master domain manager, backup master domain manager, dynamic domain manager, backup dynamic domain manager, or agent with distributed connector workstations:

Truststore

On Windows operating systems:

- <TWS_INST_DIR>\eWAS\profiles\TIPProfile\etc\TWSServerTrustFile.jks
- <TWS_INST_DIR>\eWAS\profiles\TIPProfile\etc\TWSClientTrustFile.jks

On UNIX operating systems:

- <TWS_INST_DIR>/eWAS/profiles/TIPProfile/etc/TWSServerTrustFile.jks
- <TWS_INST_DIR>/eWAS/profiles/TIPProfile/etc/TWSClientTrustFile.jks

Keystore

On Windows operating systems:

- <TWS_INST_DIR>\eWAS\profiles\TIPProfile\etc\TWSServerKeyFile.jks
- <TWS_INST_DIR>\eWAS\profiles\TIPProfile\etc\TWSClientKeyFile.jks

On UNIX operating systems:

- <TWS_INST_DIR>/eWAS/profiles/TIPProfile/etc/TWSServerKeyFile.jks
- <TWS_INST_DIR>/eWAS/profiles/TIPProfile/etc/TWSClientKeyFile.jks

where *<TWS_INST_DIR>* is the IBM Workload Scheduler installation directory.

For more information about the SSL configuration files on which the truststore and keystore information is located, see "Locating the keystore files."

Note: The default certificates are not used for the Dynamic Workload Console client authentication, which is obtained using a user ID and password.

Locating the keystore files

About this task

To locate the keystore files, run the `showSecurityProperties` utility, described in the following section: "Security properties: reference" on page 252. Then make any changes to the name, location, password of the IBM Workload Scheduler server key or truststores, you must modify the configuration files which describe them.

Client key files for all components

The client key files for IBM Workload Scheduler master are described in the file: *TWA_home*/WAS/TWSprofile/properties/ssl.client.props. The client

key files for the Dynamic Workload Console are described in the file:JazzSM_profile_dir/properties/ssl.client.props .

An example of it is as follows:

```
# KeyStore information
com.ibm.ssl.keyStoreName=ClientDefaultKeyStore
com.ibm.ssl.keyStore=/opt/ibm/TWA0/WAS/TWSprofile/etc/
                                              TWSClientKeyFile.jks
com.ibm.ssl.keyStorePassword={xor}Ozo5PiozKw\=\=
com.ibm.ssl.keyStoreType=JKS
com.ibm.ssl.keyStoreProvider=IBMJCE
com.ibm.ssl.keyStoreFileBased=true

# TrustStore information
com.ibm.ssl.trustStoreName=ClientDefaultTrustStore
com.ibm.ssl.trustStore=/opt/ibm/TWA0/WAS/TWSprofile/etc/
                                              TWSClientTrustFile.jks
com.ibm.ssl.trustStorePassword={xor}Ozo5PiozKw\=\=
com.ibm.ssl.trustStoreType=JKS
com.ibm.ssl.trustStoreProvider=IBMJCE
com.ibm.ssl.trustStoreFileBased=true
```

To modify the server key file names, paths, or passwords, modify the configuration files using the script **changeSecurityProperties** located in the *TWA_home*/TWS/wastool directory. For instructions on how to do this see "Changing the security settings" on page 443. The following is a sample of the input:

```
############################################################
SSL Panel
############################################################
alias=DefaultSSLSettings
keyFileName=${USER_INSTALL_ROOT}/etc/TWSServerKeyFile.jks
keyFilePassword=*****
keyFileFormat=JKS
trustFileName=${USER_INSTALL_ROOT}/etc/TWSServerTrustFile.jks
trustFilePassword=*****
trustFileFormat=JKS
clientAuthentication=false
securityLevel=HIGH
enableCryptoHardwareSupport=false
```

Important: The certificates for the Dynamic Workload Console have been changed and will expire after one year. To renew the certificates, follow the procedure explained in the following documentation: WebSphere Application Server section about security, renewing a certificate in SSL. .

The following table show the old and new name and path of IBM Workload Scheduler and Dynamic Workload Console certificates.

Table 68. Key and truststores

Store	Previous Certificate	Current Certificate Path
TWS server key store	TWSServerKeyFile.jks	/opt/ibm/TWA0/WAS/TWSprofile/etc/ TWSServerKeyFile.jks
TWS server truststore	TWSServerTrustFile.jks	/opt/ibm/TWA0/WAS/TWSprofile/etc/ TWSServerTrustFile.jks
TWS client key store	TWSClientKeyFile.jks	/opt/ibm/TWA0/WAS/TWSprofile/etc/ TWSClientKeyFile.jks
TWS client truststore	TWSClientTrustFile.jks	/opt/ibm/TWA0/WAS/TWSprofile/etc/ TWSClientTrustFile.jks

Table 68. Key and truststores (continued)

Store	Previous Certificate	Current Certificate Path
DWC server key store	TWSServerKeyStore.jks	*JazzSM profile dir*/config/cells/JazzSMNode01Cell/nodes/JazzSMNode01/key.p12
DWC server truststore	TWSServerTrustStore.jks	*JazzSM profile dir*/config/cells/JazzSMNode01Cell/nodes/JazzSMNode01/trust.p12
DWC client key store	TWSClientKeyStore.jks	*JazzSM profile dir*/etc/key.p12
DWC client truststore	TWSClientTrustStore.jks	*JazzSM profile dir*/etc/trust.p12

SSL connection by using your certificates

You can configure the Dynamic Workload Console to connect in SSL mode with master domain manager, backup master domain manager, dynamic domain manager, backup dynamic domain manager or agent with distributed connector by using your certificates.

About this task

You have the following environment:

Dynamic Workload Console installed on the *DWC-WKS* workstation:

- The Dynamic Workload Console is installed in the <DWC_INST_DIR> directory.
- The embedded WebSphere Application Server is installed in the <DWC_INST_DIR>\eWAS directory.

master domain manager, backup master domain manager, dynamic domain manager, backup dynamic domain manager or agent with distributed connector installed on the *TWS-WKS* workstation:

- The master domain manager, backup master domain manager, dynamic domain manager, backup dynamic domain manager or agent with distributed connector is installed in the <TWS_INST_DIR> directory.
- The embedded WebSphere Application Server is installed in the <TWS_INST_DIR>\eWas directory.

Note: The master domain manager, the backup master domain manager, the dynamic domain manager, backup dynamic domain manager or the agent with distributed connector is called *agent* with distributed connector. Also the keyword used during the keys creation is named *agent*. When you perform the procedure, you might insert a name that specifies the agent for which you are performing the procedure, i.e. *Master* for the master domain manager or *ddm* for the dynamic domain manager.

As described in Figure 6 on page 305, in the WebSphere Application Server, you must create the following keys databases:

On the agent with distributed connector instance:

- *Agent* Server key
- *Agent* Server trust
- *Agent* Client key
- *Agent* Client trust

On the Dynamic Workload Console instance:

- *DWC* Server key
- *DWC* Server trust
- *DWC* Client key
- *DWC* Client trust

and then mutually export and import the keys and enable the WebSphere Application Server to work with the new certificates.

Quick steps procedure:

1. Create the *Agent Server key* database, run 1.
2. Create the *Agent Server trust* database, run 2 on page 311.
3. Create the *Agent Client Key* database, run 3 on page 312.
4. Create the *Agent Client Trust* database, run 4 on page 313.
5. Create the *DWC Server Key* database, run 5 on page 313.
6. Create the *DWC Server Trust* database, run 6 on page 314.
7. Create the *DWC Client Key* database, run 7 on page 315.
8. Create the *DWC Client Trust* database, run 8 on page 316.
9. Import the Signed certificates into the `AgentServerTrust`, run 9 on page 317.
10. Import the Signed certificates into the`AgentClientTrust`, run 10 on page 317.
11. Import the Signed certificates into the `DWCServerTrust`, run 11 on page 318.
12. Import the Signed certificates into the `DWCClientTrust`, run 12 on page 318.
13. Configure the new server key files in the IBM Workload Scheduler agent, run 13 on page 319.
14. Configure the new client files in the IBM Workload Scheduler agent, run 14 on page 320.
15. Configure the new server key files in the Dynamic Workload Console, run 15 on page 320.
16. Configure the new client files in the Dynamic Workload Console, run 16 on page 321.

Run the following steps:

1. **Create the *Agent Server key* database**:
   a. Log on as Administrator on Windows operating systems or as root on UNIX and Linux operating systems, on the machine where you installed the *TWS-AGENT* agent.
   b. Run the `<TWS_INST_DIR>\eWas\java\jre\bin\ikeyman` command or use the `ikeyman` command provided by a Java instance on your machine.
   c. On the IBM Key Management panel, click **Key Database File** > **New**.
   d. In the New panel, enter the following information:

      `Key database type`
      Select the `JKS` type value.

      `File Name`
      Insert the `Agent Server key` value: `ServerAgentKey.jks`

      `Location`
      Insert the *<TWS_CERTS_DIR>* directory name where you want to save the `ServerAgentKey.jks` file.

   e. Click **OK**.

f. In the Password prompt panel insert the password and confirm the same password. For example *passw0rd*.

g. Click **OK**.

h. In the Key database information section on the IBM Key Management panel, you can see the `<TWS_CERTS_DIR>\ServerAgentKey.jks` just created. In the drop-down list, select **Personal certificates** and click **New Self-Signed...**.

i. Create the Self-Signed Certificate, by entering at least the following information in the Create New Self-Signed Certificate panel:

Key Label
Insert the `TWSAgentServer` value.

Version
Insert the `X509 V3` value.

Key Size
Insert the `2048` value.

Signature Algorithm
Insert the `SHA2WithRSA` value.

Common Name
Insert the `AgentServer` value.

Validity Period
Insert the `365` value.

j. Click **OK**. The *twsagentsserver* appears in the **Personal certificates** list.

k. To create the `certAgentServer.arm` certificate, select the *twsagentsserver* in the Personal Certificates list and click **Extract certificate**.

l. In the New panel, enter the following information:

Data type:
Select `Base64-encoded ASCII data`.

Certificate file name:
Insert the `certAgentServer.arm` value.

Location
Insert the *<TWS_CERTS_DIR>* directory name where you want to save the `certAgentServer.arm` file.

m. Click **OK**.

2. **Create the *Agent Server trust* database**:

a. Log on as Administrator on Windows operating systems or as root on UNIX and Linux operating systems, on the machine where you installed the *TWS-AGENT* agent.

b. Run the `<TWS_INST_DIR>\eWas\java\jre\bin\ikeyman` command or use the `ikeyman` command provided by a Java instance on your machine.

c. On the IBM Key Management panel, click **Key Database File** > **New**.

d. In the New panel, enter the following information:

Key database type
Select the `JKS` type value.

File Name
Insert the *Agent Server trust* value: `ServerAgentTrust.jks`

Location
Insert the <*TWS_CERTS_DIR*> directory name where you want to save the ServerAgentTrust.jks file.

e. Click **OK**.

f. In the Password prompt panel insert the password and confirm the same password. For example *passw0rd*.

g. Click **OK**.

h. In the drop-down list, select **Signer certificates** and Click **Add** to add the certAgentServer.arm created in 1l on page 311.

i. Enter the AgentServerTrust label for the certAgentServer.arm certificate. The *AgentServerTrust* appears in the **Signer certificates** list.

j. Click **OK**.

3. **Create the *Agent Client Key* database**:

a. Log on as Administrator on Windows operating systems or as root on UNIX and Linux operating systems, on the machine where you installed the IBM Workload Scheduler agent.

b. Run the <TWS_INST_DIR>\eWas\java\jre\bin\ikeyman command or use the ikeyman command provided by a Java instance on your machine.

c. On the IBM Key Management panel, click **Key Database File** > **New**.

d. In the New panel, enter the following information:

Key database type
Select the JKS type value.

File Name
Insert the *Agent Client Key* value: ClientAgentKey.jks

Location
Insert the <*TWS_CERTS_DIR*> directory name where you want to save the ClientAgentKey.jks file.

e. Click **OK**.

f. In the Password prompt panel insert the password and confirm the same password. For example *passw0rd*.

g. Click **OK**.

h. In the Key database information section on the IBM Key Management panel, you can see the <TWS_CERTS_DIR>\ClientAgentKey.jks just created. In the drop-down list, select **Personal certificates** and click **New Self-Signed...**.

i. Create the Self-Signed Certificate, by entering at least the following information in the Create New Self-Signed Certificate panel:

Key Label
Insert the TWSAgentClient value.

Version
Insert the X509 V3 value.

Key Size
Insert the 2048 value.

Signature Algorithm
Insert the SHA2WithRSA value.

Common Name
Insert the AgentClient value.

Validity Period
Insert the 365 value.

j. Click **OK**. The *twsagentsClient* appears in the **Personal certificates** list.

k. To create the certAgentClient.arm certificate, select the *twsagentsClient* in the Personal Certificates list and click **Extract certificate**.

l. In the New panel, enter the following information:

Data type:
Select Base64-encoded ASCII data.

Certificate file name:
Insert the certAgentClient.arm value.

Location
Insert the <*TWS_CERTS_DIR*> directory name where you want to save the certAgentClient.arm file.

m. Click **OK**.

4. **Create the *Agent Client Trust* database**:

a. Log on as Administrator on Windows operating systems or as root on UNIX and Linux operating systems, on the machine where you installed the *TWS-AGENT* agent.

b. Run the <TWS_INST_DIR>\eWas\java\jre\bin\ikeyman command or use the ikeyman command provided by a Java instance on your machine.

c. On the IBM Key Management panel, click **Key Database File** > **New**.

d. In the New panel, enter the following information:

Key database type
Select the JKS type value.

File Name
Insert the *Agent Client Trust* value: ClientAgentTrust.jks

Location
Insert the <*TWS_CERTS_DIR*> directory name where you want to save the ClientAgentTrust.jks file.

e. Click **OK**.

f. In the Password prompt panel insert the password and confirm the same password. For example *passw0rd*.

g. Click **OK**.

h. In the drop-down list, select **Signer certificates** and Click **Add** to add the certAgentClient.arm created in 3l.

i. Enter the *ClientAgentTrust* label for the certAgentClient.arm certificate. The *ClientAgentTrust* appears in the **Signer certificates** list.

j. Click **OK**.

5. **Create the *DWC Server Key* database**:

a. Log on as Administrator on Windows operating systems or as root on UNIX and Linux operating systems, on the machine where you installed the *DWC* Dynamic Workload Console.

b. Run the <DWC_INST_DIR>\eWas\java\jre\bin\ikeyman command or use the ikeyman command provided by a Java instance on your machine.

c. On the IBM Key Management panel, click **Key Database File** > **New**.

d. In the New panel, enter the following information:

Key database type
Select the JKS type value.

File Name
Insert the *DWC Server Key* value: ServerDWCKey.jks

Location
Insert the <*DWC_CERTS_DIR*> directory name where you want to save the ServerDWCKey.jks file.

e. Click **OK**.

f. In the Password prompt panel insert the password and confirm the same password. For example *passw0rd*.

g. Click **OK**.

h. In the Key database information section on the IBM Key Management panel, you can see the <TWS_CERTS_DIR>\ServerDWCKey.jks just created. In the drop-down list, select **Personal certificates** and click **New Self-Signed...**.

i. Create the Self-Signed Certificate, by entering at least the following information in the Create New Self-Signed Certificate panel:

Key Label
Insert the TWSDWCServer value.

Version
Insert the X509 V3 value.

Key Size
Insert the 2048 value.

Signature Algorithm
Insert the SHA2WithRSA value.

Common Name
Insert the DWCServer value.

Validity Period
Insert the 365 value.

j. Click **OK**. The *twsDWCserver* appears in the **Personal certificates** list.

k. To create the certDWCServer.arm certificate, select the *twsDWCsserver* in the Personal Certificates list and click **Extract certificate**.

l. In the New panel, enter the following information:

Data type:
Select Base64-encoded ASCII data.

Certificate file name:
Insert the certDWCServer.arm value.

Location
Insert the <*DWC_CERTS_DIR*> directory name where you want to save the certDWCServer.arm file.

m. Click **OK**.

6. **Create the *DWC Server Trust* database**:

a. Log on as Administrator on Windows operating systems or as root on UNIX and Linux operating systems, on the machine where you installed the *DWC* Dynamic Workload Console.

b. Run the <DWC_INST_DIR>\eWas\java\jre\bin\ikeyman command or use the ikeyman command provided by a Java instance on your machine.

c. On the IBM Key Management panel, click **Key Database File** > **New**.
d. In the New panel, enter the following information:

Key database type
Select the JKS type value.

File Name
Insert the *DWC Server Trust* value: ServerDWCTrust.jks

Location
Insert the <*DWC_CERTS_DIR*> directory name where you want to save the ServerDWCTrust.jks file.

e. Click **OK**.
f. In the Password prompt panel insert the password and confirm the same password. For example *passw0rd*.
g. Click **OK**.
h. In the drop-down list, select **Signer certificates** and Click **Add** to add the certDWCServer.arm created in 5l on page 314.
i. Enter the *DWCServerTrust* label for the certDWCServer.arm certificate. The certDWCServer appears in the **Signer certificates** list.
j. Click **OK**.

7. **Create the *DWC Client Key* database**:
a. Log on as Administrator on Windows operating systems or as root on UNIX and Linux operating systems, on the machine where you installed the *DWC* Dynamic Workload Console.
b. Run the <DWC_INST_DIR>\eWas\java\jre\bin\ikeyman command or use the ikeyman command provided by a Java instance on your machine.
c. On the IBM Key Management panel, click **Key Database File** > **New**.
d. In the New panel, enter the following information:

Key database type
Select the JKS type value.

File Name
Insert the *DWC Client Key* value: ClientDWCKey.jks

Location
Insert the <*DWC_CERTS_DIR*> directory name where you want to save the ClientDWCKey.jks file.

e. Click **OK**.
f. In the Password prompt panel insert the password and confirm the same password. For example *passw0rd*.
g. Click **OK**.
h. In the Key database informationsection on the IBM Key Management panel, you can see the <DWC_CERTS_DIR>\ClientDWCKey.jks just created. In the drop-down list, select **Personal certificates** and click **New Self-Signed...**.
i. Create the Self-Signed Certificate, by entering at least the following information in the Create New Self-Signed Certificate panel:

Key Label
Insert the TWSDWCClient value.

Version
Insert the X509 V3 value.

Key Size
Insert the 2048 value.

Signature Algorithm
Insert the SHA2WithRSA value.

Common Name
Insert the DWCClient value.

Validity Period
Insert the 365 value.

j. Click **OK**. The *twsDWCClient* appears in the **Personal certificates** list.

k. To create the certDWCClient.arm certificate, select the *twsDWCClient* in the Personal Certificates list and click **Extract certificate**.

l. In the New panel, enter the following information:

Data type:
Select Base64-encoded ASCII data.

Certificate file name:
Insert the certDWCClient.arm value.

Location
Insert the <*TWS_CERTS_DIR*> directory name where you want to save the certDWCClient.arm file.

m. Click **OK**.

8. **Create the *DWC Client Trust* database**:

a. Log on as Administrator on Windows operating systems or as root on UNIX and Linux operating systems, on the machine where you installed the *DWC* Dynamic Workload Console.

b. Run the <DWC_INST_DIR>\eWas\java\jre\bin\ikeyman command or use the ikeyman command provided by a Java instance on your machine.

c. On the IBM Key Management panel, click **Key Database File** > **New**.

d. In the New panel, enter the following information:

Key database type
Select the JKS type value.

File Name
Insert the *DWC Client Trust* value: ClientDWCTrust.jks

Location
Insert the <*DWC_CERTS_DIR*> directory name where you want to save the ClientDWCTrust.jks file.

e. Click **OK**.

f. In the Password prompt panel insert the password and confirm the same password. For example *passw0rd*.

g. Click **OK**.

h. In the drop-down list, select **Signer certificates** and Click **Add** to add the certDWCClient.arm created in 7l.

i. Enter the *DWCClientTrust* label for the certDWCClient.arm certificate. The *DWCClientTrust* appears in the **Signer certificates** list.

j. Click **OK**.

9. **Import the <TWS_CERTS_DIR>\certAgentClient.arm, <TWS_CERTS_DIR>\certAgentServer.arm, and <DWC_CERTS_DIR>\certDWCServer.arm certificates in the AgentServerTrust Signed certificates as described in Figure 6 on page 305**:
   a. Copy the <DWC_CERTS_DIR>\certDWCServer.arm certificate from the *DWC-WKS* workstation to the *TWS-WKS* workstation in the <TWS_CERTS_DIR> directory.
   b. Click **Open** to open the AgentServerTrust signed certificates created in the 2i on page 312.
   c. In the Open panel, enter the following information:

      Key database type
      Select the JKS type value.

      File Name
      Enter the <TWS_CERTS_DIR>\ServerAgentTrust.jks

      Location
      Insert the <*TWS_CERTS_DIR*> directory name.
   d. Click **OK**.
   e. In the Password prompt panel insert the password and confirm the same password. For example *passw0rd*.
   f. Click **OK**.
   g. Select the AgentServerTrust signed certificates created in the 2i on page 312.
   h. Click **Add** to add the <TWS_CERTS_DIR>\certAgentClient.arm created in 3l on page 313.
   i. Click **OK**.
   j. Click **Add** to add the <TWS_CERTS_DIR>\certAgentServer.arm created in 1l on page 311.
   k. Click **OK**.
   l. Click **Add** to add the <DWC_CERTS_DIR>\certDWCServer.arm created in 5l on page 314.
   m. Click **OK**.
10. **Import the <TWS_CERTS_DIR>\certAgentServer.arm certificate in the AgentClientTrust Signed certificates as described in Figure 6 on page 305**, by performing the following steps:
   a. Click **Open** to open the AgentClientTrust signed certificates created in the 4i on page 313.
   b. In the Open panel, enter the following information:

      Key database type
      Select the JKS type value.

      File Name
      Enter the <TWS_CERTS_DIR>\ClientAgentTrust.jks

      Location
      Insert the <*TWS_CERTS_DIR*> directory name.
   c. Click **OK**.
   d. In the Password prompt panel insert the password and confirm the same password. For example *passw0rd*.
   e. Click **OK**.
   f. In the drop-down list, select **Signed certificates**.

g. Select the AgentClientTrust signed certificates created in 4i on page 313.
h. Click **Add** to add the <TWS_CERTS_DIR>\certAgentServer.arm created in 1l on page 311.
i. Click **OK**.

11. **Import the <TWS_CERTS_DIR>\certAgentServer.arm, <DWC_CERTS_DIR>\certDWCServer.arm and <DWC_CERTS_DIR>\certDWCClient.arm certificates in the DWCServerTrust Signed certificates as described in Figure 6 on page 305**, by performing the following steps:
    a. Copy the <TWS_CERTS_DIR>\certAgentServer.arm certificate from the *TWS-WKS* workstation to the *DWC-WKS* workstation in the <DWC_CERTS_DIR> directory.
    b. Click **Open** to open the DWCServerTrust signed certificates created in the 6i on page 315.
    c. In the Open panel, enter the following information:

       Key database type
       Select the JKS type value.

       File Name
       Enter the <DWC_CERTS_DIR>\ServerDWCTrust.jks

       Location
       Insert the <*DWC_CERTS_DIR*> directory name.
    d. Click **OK**.
    e. In the Password prompt panel insert the password and confirm the same password. For example *passw0rd*.
    f. Click **OK**.
    g. In the drop-down list, select **Signer certificates**.
    h. Select the DWCServerTrust signed certificates created in the 6i on page 315.
    i. Click **Add** to add the <DWC_CERTS_DIR>\certAgentServer.arm created in 1l on page 311.
    j. Click **OK**.
    k. Select the DWCServerKey signed certificates created in the 5l on page 314.
    l. Click **Add** to add the <DWC_CERTS_DIR>\certDWCServer.arm created in 5l on page 314.
    m. Click **OK**.
    n. Click **Add** to add the <DWC_CERTS_DIR>\certDWCClient.arm created in 7l on page 316.
    o. Click **OK**.
12. **Import the <DWC_CERTS_DIR>\certDWCServer.arm certificate in the DWCClientTrust Signed certificates** as described in Figure 6 on page 305, by performing the following steps:
    a. Click **Open** to open the DWCClientTrust signed certificates created in the 8i on page 316.
    b. In the Open panel, enter the following information:

       Key database type
       Select the JKS type value.

       File Name
       Enter the <DWC_CERTS_DIR>\ClientDWCTrust.jks

       Location
       Insert the <*DWC_CERTS_DIR*> directory name.

c. Click **OK**.

d. In the Password prompt panel insert the password and confirm the same password. For example *passw0rd*.

e. Click **OK**.

f. In the drop-down list, select **Signer certificates**.

g. Select the `DWCClientTrust` signed certificates created in the 8i on page 316.

h. Click **Add** to add the `<DWC_CERTS_DIR>\certDWCServer.arm` created in 5l on page 314.

i. Click **OK**.

13. **Configure the new server key files in the IBM Workload Scheduler agent with distributed connector**:

a. Stop the WebSphere Application Server on the IBM Workload Scheduler agent with distributed connector. For more information about this utility, see *Administration Guide> Administrative tasks > Application server tasks*.

b. Run the following script:

On Windows operating systems:

```
showSecurityProperties.bat > My_Security.prop
```

On UNIX and Linux operating systems:

```
showSecurityProperties.sh > My_Security.prop
```

c. In the `My_Security.prop` file SSL Panel section , insert the `keyFileName` name that you created in 1d on page 310 and `trustFileName` name that you created in 2d on page 311:

Note: Use / for Windows and UNIX operating systems.

```
#############################################################
SSL Panel
#############################################################
alias=NodeDefaultSSLSettings
keyFileName=
<TWS_CERTS_DIR>/ServerAgentKey.jks
keyFilePassword=*****
keyFileFormat=JKS
trustFileName=
<TWS_CERTS_DIR>/ServerAgentTrust.jks
trustFilePassword=*****
trustFileFormat=JKS
clientAuthentication=false
securityLevel=HIGH
enableCryptoHardwareSupport=false
```

Note:

- On Windows and UNIX operating systems, use the / in the `keyfilename` and `trustFilename` path.
- Encrypt the password using the **`encryptProfileProperties`** utility. For more information about this utility, see *Administration Guide> Administrative tasks > Application server tasks >encrypting the profile properties files for details on how to encrypt profile properties*

d. Modify the Security properties, by running the following script:

On Windows operating systems:

```
changeSecurityProperties.bat  My_Security.prop
```

On UNIX and Linux operating systems:

```
changeSecurityProperties.sh  My_Security.prop
```

14. **Configure the new client files in the IBM Workload Scheduler agent with distributed connector**:
    a. Locate the following file:

       On Windows operating systems:
       <TWS_INST_DIR>\eWas\profiles\TIPProfile\properties\ssl.client.props

       On UNIX and Linux operating systems:
       <TWS_INST_DIR>/eWas/profiles/TIPProfile/properties/ssl.client.props

    b. In the ssl.client.props file, modify the KeyStore information and TrustStore information section, by insert the following values:

```
# KeyStore information
com.ibm.ssl.keyStoreName=ClientDefaultKeyStore
com.ibm.ssl.keyStore=<TWS_CERTS_DIR>/ClientAgentKey.jks
com.ibm.ssl.keyStorePassword=passw0rd
com.ibm.ssl.keyStoreType=JKS
com.ibm.ssl.keyStoreProvider=IBMJCE
com.ibm.ssl.keyStoreFileBased=true

# TrustStore information
com.ibm.ssl.trustStoreName=ClientDefaultTrustStore
com.ibm.ssl.trustStore=<TWS_CERTS_DIR>/ClientAgentTrust.jks
com.ibm.ssl.trustStorePassword=passw0rd
com.ibm.ssl.trustStoreType=JKS
com.ibm.ssl.trustStoreProvider=IBMJCE
com.ibm.ssl.trustStoreFileBased=true
com.ibm.ssl.trustStoreReadOnly=false
```

       where

       com.ibm.ssl.keyStore
       Insert the <TWS_CERTS_DIR>/ClientAgentKey.jks file that you generated in 3d on page 312.

       com.ibm.ssl.keyStorePassword
       Insert the password value that you used in 3f on page 312.

       com.ibm.ssl.trustStore
       Insert the <TWS_CERTS_DIR>/ClientAgentTrust.jks file that you generated in 4d on page 313.

       com.ibm.ssl.trustStorePassword
       Insert the password value that you used in 4f on page 313.

       Note:
       - On Windows and UNIX operating systems, use the / in the keyfilename and trustFilename path.
       - Encrypt the password using the **encryptProfileProperties** utility. For more information about this utility, see *Administration Guide> Administrative tasks > Application server tasks >encrypting the profile properties files for details on how to encrypt profile properties*

    c. Start the WebSphere Application Server on the IBM Workload Scheduler agent with distributed connector. For more information about this utility, see *Administration Guide> Administrative tasks > Application server tasks*.

15. **Configure the new server key files in the Dynamic Workload Console**:
    a. Stop the WebSphere Application Server on the Dynamic Workload Console. For more information about this utility, see *Administration Guide> Administrative tasks > Application server tasks*.

b. Run the following script:

On Windows operating systems:

```
showSecurityProperties.bat > My_Security.prop
```

On UNIX and Linux operating systems:

```
showSecurityProperties.sh > My_Security.prop
```

c. In the `My_Security.prop` file SSL Panel section , insert the `keyFileName` name that you created in 5d on page 313 and `trustFileName` name that you created in 6d on page 315:

```
############################################################
SSL Panel
############################################################
alias=NodeDefaultSSLSettings
keyFileName=
<TWS_CERTS_DIR>/ServerDWCKey.jks
keyFilePassword=passw0rd
keyFileFormat=JKS
trustFileName=
<TWS_CERTS_DIR>/ServerDWCTrust.jks
trustFilePassword=passw0rd
trustFileFormat=JKS
clientAuthentication=false
securityLevel=HIGH
enableCryptoHardwareSupport=false
```

Note:

- On Windows and UNIX operating systems, use the / in the `keyfilename` and `trustFilename` path.
- Encrypt the password using the **encryptProfileProperties** utility. For more information about this utility, see *Administration Guide> Administrative tasks > Application server tasks >encrypting the profile properties files for details on how to encrypt profile properties*

d. Modify the Security properties, by running the following script:

On Windows operating systems:

```
changeSecurityProperties.bat  My_Security.prop
```

On UNIX and Linux operating systems:

```
changeSecurityProperties.sh  My_Security.prop
```

16. **Configure the new client files in the Dynamic Workload Console**:

a. Locate the following file:

On Windows operating systems:

```
<DWC_INST_DIR>\eWas\profiles\TIPProfile\properties\
ssl.client.props
```

On UNIX and Linux operating systems:

```
<DWC_INST_DIR>/eWas/profiles/TIPProfile/properties/
ssl.client.props
```

b. In the `ssl.client.props` file, modify the KeyStore information and TrustStore information sections, by insert the following values:

```
# KeyStore information
com.ibm.ssl.keyStoreName=ClientDefaultKeyStore
com.ibm.ssl.keyStore=<DWC_CERTS_DIR>/ClientDWCKey.jks
com.ibm.ssl.keyStorePassword=passw0rd
com.ibm.ssl.keyStoreType=JKS
com.ibm.ssl.keyStoreProvider=IBMJCE
com.ibm.ssl.keyStoreFileBased=true

# TrustStore information
```

```
com.ibm.ssl.trustStoreName=ClientDefaultTrustStore
com.ibm.ssl.trustStore=<DWC_CERTS_DIR>/ClientDWCTrust.jks
com.ibm.ssl.trustStorePassword=passw0rd
com.ibm.ssl.trustStoreType=JKS
com.ibm.ssl.trustStoreProvider=IBMJCE
com.ibm.ssl.trustStoreFileBased=true
com.ibm.ssl.trustStoreReadOnly=false
```

where

com.ibm.ssl.keyStore
Insert the `<DWC_CERTS_DIR>/ClientDWCKey.jks` file that you generated in 7d on page 315.

com.ibm.ssl.keyStorePassword
Insert the password value that you used in 7f on page 315.

com.ibm.ssl.trustStore
Insert the `<DWC_CERTS_DIR>/ClientDWCTrust.jks` file that you generated in 8d on page 316.

com.ibm.ssl.trustStorePassword
Insert the password value that you used in 8f on page 316.

Note:

- On Windows and UNIX operating systems, use the / in the `keyfilename` and `trustFilename` path.
- Encrypt the password using the **encryptProfileProperties** utility. For more information about this utility, see *Administration Guide> Administrative tasks > Application server tasks >encrypting the profile properties files for details on how to encrypt profile properties*

c. Start the WebSphere Application Server on the Dynamic Workload Console. For more information about this utility, see *Administration Guide> Administrative tasks > Application server tasks*.

Scenario: Connection between dynamic agents and the master domain manager or dynamic domain manager

The default certificates provided during the IBM Workload Scheduler installation process, ensure the secure connection between the following components:

Master domain manager and dynamic domain manager or backup dynamic domain manager:

The connection to the broker server installed with the dynamic domain manager requires the use of certificates from a certificate authority to provide authentication. In addition, the master domain managers and the backup master domain managers that communicate with the dynamic domain manager or its backup must be defined on the related broker server to ensure role-based authorization.

The dynamic domain manager communicates with any of the following components:

- Master domain manager
- Backup master domain manager (if any)
- Backup dynamic domain manager (if any)

Master domain manager or dynamic domain manager and dynamic agents:

The dynamic agent communicates with any of the following components:

- Master domain manager
- Backup master domain manager (if any)
- Dynamic domain manager (if any)
- Backup dynamic domain manager (if any)

Communication between dynamic agents and a master domain manager, or a dynamic domain manager to which they are registered, is by default in https. This communication uses the product default certificates.

ResourceCLI command line and the Broker Server installed on the master domain manager

You can enable the communication between the **ResourceCLI** command line installed on the dynamic domain manager and the Broker Server installed on the master domain manager by using the default certificates or by using your own certificates. See "Customizing the SSL connection between a master domain manager and the resource command line" on page 325.

Customizing the SSL connection between a master domain manager and a dynamic domain manager or its backup by using your certificates

Customizing the SSL connection between a master domain manager and a dynamic domain manager or its backup by using your certificates.

About this task

The connection to the broker server installed with the dynamic domain manager requires the use of certificates from a certificate authority to provide authentication. In addition, the master domain managers and the backup master domain managers that communicate with the dynamic domain manager or the backup dynamic domain managers must be defined on the related broker server to ensure role-based authorization.

The examples in this section refer to a dynamic domain manager that communicates with a master domain manager, but the same configuration applies also when the dynamic domain manager communicates with any of the following components:

- Master domain manager
- Backup master domain manager (if any)
- Backup dynamic domain manager (if any)

If you use the default certificates installed with the product, the communication between all the components is automatically achieved.

When you install IBM Workload Scheduler, the default certificates provided ensure correct authentication and role-based authorization between the components. The default value for the certificate is `Server` on the master domain manager.

If you plan to use your certificates rather than the default ones, to enable the communication between components follow the procedure described in the section below.

For example, the following procedure enables communication between a master domain manager and a dynamic domain manager.

Procedure

Procedure to enable the communication between a master domain manager and a dynamic domain manager:

1. Modify the certificate on the master domain manager. For example, this procedure assumes that the common name present in the certificate on the master domain manager is `mdm1`.
2. Deploy the certificate to the master domain manager and to the dynamic domain manager, as described in Chapter 7, "Setting connection security," on page 303.
3. Modify the list of common names on the dynamic domain manager, as follows:
   a. Browse to *TWA_home*/TDWB/config.
   b. Open the file `BrokerWorkstation.properties`.
   c. In the option `Broker.AuthorizedCNs`, define the common name for the authorized master domain manager. In this example

      ```
      Broker.AuthorizedCNs=mdm1
      ```

      If you want to enable the communication with more than one master domain manager, separate each value with a semicolon. For example, you can define the following list:

      ```
      Broker.AuthorizedCNs=mdm;mdm1;mdm2
      ```

      This list ensures that all master domain managers with those common names can connect to the dynamic domain manager.
4. Stop and start the dynamic domain manager to make the change effective, as follows:
   a. Use the wastool **stopBrokerApplication.sh** on UNIX and Linux or **stopBrokerApplication.bat** on Windows:

      ```
      stopBrokerApplication -user username
      -password password [-port portnumber]
      ```

      where *username* and *password* are the credentials used at installation. The parameter *portnumber* is optional; if it is not specified, the default is used.
   b. Use the wastool **startBrokerApplication.sh** on UNIX and Linux or **startBrokerApplication.bat** on Windows:

      ```
      startBrokerApplication -user username -password password [-port portnumber]
      ```

      where *username* and *password* are the credentials used at installation. The parameter *portnumber* is optional. If it is not specified, the default is used.

For more information, see "BrokerWorkstation.properties file" on page 79 and "Starting, stopping, and displaying dynamic workload broker status" on page 445.

Customizing the SSL connection between dynamic agents and a master domain manager or a dynamic domain manager using your certificates

Customizing the SSL connection between a master domain manager or a dynamic domain manager and dynamic agents connected to it using your certificates.

About this task

Communication between dynamic agents and a master domain manager, or a dynamic domain manager to which they are registered, is by default in https. This

communication uses the product default certificates. If you want to use your own customized certificates for this communication because you customized the master domain manager or the dynamic domain manager certificates you must customize the agent certificates and configuration. To enable the communication between dynamic agents and a master domain manager or a dynamic domain manager, perform the following steps:

1. Generate a `.kdb` CMS key store file. This file must contain a private key trusted by the master domain manager or the dynamic domain manager to which the agent is registered, and the master domain manager or the dynamic domain manager public key so that the agent can trust the them. The private key present in TWSClientKeyStore.kdb on the agent must be trusted by the master domain manager, therefore the agent's public certificate must be stored in `TWSServerTrustFile.jks` in the master domain manager.

   Note: If the private key is provided by a Certificate Authority, the entire certificate chain must be stored in the **TWSServerTrustFile.jks** file. For details, see the Certificate Authority documentation.
   The master domain manager's private key must also be trusted by the agent, therefore the master domain manager's public certificate must be stored in **TWSClientKeyStore.kdb** in the master domain manager.
2. Save the password of the key store in a stash file that has the same name as the file you generated in Step 1 and with extension `.sth`.
3. Open the `ita.ini` agent configuration file and set the values specific for your environment to the following properties:

   ```
   cert_label=<label_agent_private_key>
   key_db_name=<file_name>
   key_repository_dir=<directory>
   ```

   Where:

   label_agent_private_key
   Specify the label of the agent private key that you want to use for the communication. The default is **client**.

   file_name
   Specify the name of the file without the extension. The default value is **TWSClientKeyStore**.

   directory
   Specify the directory that contains the files generated in Step 1 and in Step 2. The default path is `/opt/IBM/TWA/TWS/ITA/cpa/ita/cert`.
4. Stop the IBM i agent by using the following command:

   ```
   ShutDownLwa
   ```
5. Start the IBM i agent by using the following command:

   ```
   StartUpLwa
   ```

Customizing the SSL connection between a master domain manager and the resource command line

Customizing the SSL connection between a master domain manager and the resource command line.

About this task

The communication between the resource command line and the master domain manager is by default in http. If you want to use https, you can use the default certificates or your certificates.

If you want to use the default certificates, run the following procedure:

1. Find the value you set for the port of the WebSphere Application Server of the master domain manager in the **httpsPort** property, by running the **showHostProperties** wastool. The default value is **31116**.
2. Open the `TWS/TDWB_CLI/config/CLIConfig.properties` file.
3. Set the **ITDWBServerSecurePort** property to this value. For example, if you used the **31116** port, write:

   ```
   ITDWBServerSecurePort=31116
   ```
4. Set the **use_secure_connection** property to **true**, write.

   ```
   use_secure_connection=true
   ```

   In this way you use the default certificates provided with the product that are stored in the following keyStore and trustStore on the agent on which you are using the resource command line:

   ```
   keyStore=TWS_inst_dir/TWS/TDWB_CLI/certs/TWSClientKeyFile.jks
   trustStore=TWS_inst_dir/TWS/TDWB_CLI/certs/TWSClientTrustFile.jks
   ```

If you want to use your own certificates, run the Steps from 1 to 4 and then perform the following step:

- Substitute the default certificates present on the agent with the customized certificates present on the master domain manager. The master domain manager certificates are located in the *<WAS_profile_path>*`/etc` directory where the default value for *<WAS_profile_path>* is *<TWA_home>*`/WAS/TWSprofile`. Ensure that keyStore and trustStore properties on the agent point to the correct certificates. For example, if you stored the master domain manager in the `tmp` directory in the agent keyStore and trustStore, write:

  ```
  keyStore=tmp/TWS/TDWB_CLI/certs/TWSClientKeyFile.jks
  trustStore=tmp/TWS/TDWB_CLI/certs/TWSClientTrustFile.jks
  ```

Scenario: SSL Communication across the IBM Workload Scheduler network

You can enable the SSL connection using OpenSSL Toolkit for the following components:

- Master domain manager and its domain managers
- Master domain manager and fault-tolerant agents in the master domain
- Master domain manager and backup master domain manager
- Domain manager and fault-tolerant agents that belong to that domain

The default certificates are located in the `<INSTALL_DIR>\TWS\ssl\OpenSSL` directory.

Using SSL for netman and conman

IBM Workload Scheduler provides a secure, authenticated, and encrypted connection mechanism for communication across the network topology. This mechanism is based on the Secure Sockets Layer (SSL) protocol and uses the OpenSSL Toolkit, which is automatically installed with IBM Workload Scheduler.

The SSL protocol is based on a private and public key methodology. SSL provides the following authentication methods:

CA trusting only
Two workstations trust each other if each receives from the other a certificate that is signed or is trusted. That is, if the CA certificate is in the list of trusted CAs on each workstation. With this authentication level, a workstation does not perform any additional checks on certificate content, such as the distinguished name. Any signed or trusted certificate can be used to establish an SSL session. See "Setting local options" on page 34 for a definition of the **caonly** option used by the **ssl auth mode** keyword.

Check if the distinguished name matches a defined string
Two workstations trust each other if, after receiving a trusted or signed certificate, each performs a further check by extracting the distinguished name from the certificate and comparing it with a string that was defined in its local options file. See "Setting local options" on page 34 for a definition of the **string** option.

Check if the distinguished name matches the workstation name
Two workstations trust each other if, after receiving a signed or trusted certificate, each performs a further check by extracting the distinguished name from the certificate and comparing it with the name of the workstation that sent the certificate. See "Setting local options" on page 34 for a definition of the **cpu** option.

To provide SSL security for a domain manager attached to z/OS in an end-to-end connection, configure the OS/390® Cryptographic Services System SSL in the IBM Workload Scheduler code that runs in the OS/390 USS UNIX shell in the IBM Workload Scheduler for z/OS server address space. See the IBM Workload Scheduler z/OS documentation.

When configuring SSL you can:

Use the same certificate for the entire network
If the workstations are configured with CA trusting only, they accept connections with any other workstation that sends a signed or trusted certificate. To enforce the authentication you define a name or a list of names that must match the contents of the certificate distinguished name (DN) field in the `localopts` file of each workstation.

Use a certificate for each domain
Install private keys and signed certificates for each domain in the network. Then, configure each workstation to accept a connection only with partners that have a particular string of the certificate DN field in the `localopts` file of each workstation.

Use a certificate for each workstation
Install a different key and a signed certificate on each workstation and add a Trusted CA list containing the CA that signed the certificate. Then, configure each workstation to accept a connection only with partners that have their workstation name specified in the `Symphony` file recorded in the DN field of the certificate.

Setting up private keys and certificates

About this task

To use SSL authentication on a workstation, you need to create and install the following:

- The private key and the corresponding certificate that identify the workstation in an SSL session.
- The list of certificate authorities that can be trusted by the workstation.

Use the **openssl** command line utility to:

- Create a file containing pseudo random generated bytes (TWS.rnd). This file is needed on some operating systems for SSL to function correctly.
- Create a private key.
- Save the password you used to create the key into a file.
- Create a Certificate Signing Request.
- Send this Certificate Signing Request (CSR) to a Certifying Authority (CA) for signing, or:
  - Create your own Certificate Authority (CA)
  - Create a self-signed CA Certificate (X.509 structure) with the RSA key of your own CA
  - Use your own Certificate Authority (CA) to sign and create real certificates

These actions will produce the following files that you will install on the workstation(s):

- A private key file (for example, TWS.key). This file should be protected, so that it is not stolen to use the workstation's identity. You should save it in a directory that allows read access to the TWS user of the workstation, such as *TWA_home*/TWS/ssl/TWS.key.
- The corresponding certificate file (for example, TWS.crt). You should save it in a directory that allows read access to the TWS user of the workstation, such as *TWA_home*/TWS/ssl/TWS.crt.
- A file containing a pseudo-random generated sequence of bytes. You can save it in any directory that allows read access to the TWS user of the workstation, such as *TWA_home*/TWS/ssl/TWS.rnd.

In addition, you should create the following:

- A file containing the password used to encrypt the private key. You should save it in a directory that allows read access to the TWS user of the workstation, such as *TWA_home*/TWS/ssl/TWS.sth.
- The certificate chain file. It contains the concatenation of the PEM-encoded certificates of certification authorities which form the certificate chain of the workstation's certificate. This starts with the issuing CA certificate of the workstation's certificate and can range up to the root CA certificate. Such a file is simply the concatenation of the various PEM-encoded CA certificate files, usually in certificate chain order.
- The trusted CAs file. It contains the trusted CA certificates to use during authentication. The CAs in this file are also used to build the list of acceptable client CAs passed to the client when the server side of the connection requests a client certificate. This file is simply the concatenation of the various PEM-encoded CA certificate files, in order of preference.

Creating Your Own Certification Authority
About this task

If you are going to use SSL authentication within your company's boundaries and not for outside internet commerce, you might find it simpler to create your own certification authority (CA) to trust all your IBM Workload Scheduler installations. To do so, follow the steps listed below.

Note: In the following steps, the names of the files created during the process TWS and TWSca are sample names. You can use your own names, but keep the same file extensions.

1. Choose a workstation as your CA root installation.
2. Type the following command from the SSL directory to initialize the pseudo random number generator, otherwise subsequent commands might not work.
   - On UNIX:

     ```
     $ openssl rand -out TWS.rnd -rand ./openssl 8192
     ```
   - On Windows:

     ```
     $ openssl rand -out TWS.rnd -rand ./openssl.exe 8192
     ```
3. Type the following command to create the CA private key:

   ```
   $ openssl genrsa -out TWSca.key 2048
   ```
4. Type the following command to create a self-signed CA Certificate (X.509 structure):

   ```
   $ openssl req -new -x509 -days 365 -key TWSca.key -out TWSca.crt -config ./
     openssl.cnf
   ```

Now you have a certification authority that you can use to trust all of your installations. If you want, you can create more than one CA.

Creating private keys and certificates
About this task

The following steps explain how to create one key and one certificate. You can decide to use one key and certificate pair for the entire network, one for each domain, or one for each workstation. The steps below assume that you will be creating a key and certificate pair for each workstation and thus the name of the output files created during the process has been generalized to *workstationname*.

On each workstation, perform the following steps to create a private key and a certificate:

1. Enter the following command from the SSL directory to initialize the pseudo random number generator, otherwise subsequent commands might not work.
   - On Windows operating systems:

     ```
     $ openssl rand -out workstationname.rnd -rand ./openssl.exe 8192
     ```
   - On UNIX and Linux operating systems :

     ```
     $ openssl rand -out workstationname.rnd -rand ./openssl 8192
     ```
2. Enter the following command to create the private key (this example shows triple-DES encryption):

   ```
   $ openssl genrsa -des3 -out workstationname.key 2048
   ```

   Then, save the password that was requested to encrypt the key in a file named *workstationname*.pwd.

Note: Verify that file *workstationname*.pwd contains just the characters in the password. For instance, if you specified the word *maestro* as the password, your *workstationname*.pwd file should not contain any CR or LF characters at the end (it should be 7 bytes long).

3. Create stash file, you can choose to create a stash file or encrypt you password file:

   stash file

   Enter the following command to save your password, encoding it in base64 into the appropriate stash file:

   ```
   $ openssl base64 -in workstationname.pwd -out workstationname.sth
   ```

   You can then delete file *workstationname*.pwd.

   encrypted password file

   Run the following command to save your encrypted password, encoding it in base64:

   ```
   $ conman crypt workstationname.pwd
   ```

   Example: If you have the *workstationname*.pwd that contains the string *secreat* that is the password you set, after you run the `$ conman crypt workstationname.pwd`, your *workstationname*.pwd file contains the string *{3DES}poh56FeTy+=/jhtf2djur* that is the encrypted password.

4. Enter the following command to create a certificate signing request (CSR):

   ```
   $ openssl req -new -key workstationname.key -out workstationname.csr
      -config ./openssl.cnf
   ```

   Some values-such as company name, personal name, and more- will be requested at screen. For future compatibility, you might specify the workstation name as the distinguished name.

5. Send the *workstationname*.csr file to your CA in order to get the matching certificate for this private key.

   Using its private key (TWSca.key) and certificate (TWSca.crt), the CA will sign the CSR (*workstationname*.csr) and create a signed certificate (*workstationname*.crt) with the following command:

   ```
   $ openssl x509 -req -CA TWSca.crt -CAkey TWSca.key -days 365
      -in workstationname.csr   -out workstationname.crt -CAcreateserial
   ```

6. Distribute to the workstation the new certificate *workstationname*.crt and the public CA certificate TWSca.crt.

The table below summarizes which of the files created during the process have to be set as values for the workstation's local options.

Table 69. Files for Local Options

Local option	File
SSL key	*workstationname*.key
SSL certificate	*workstationname*.crt
SSL key pwd	*workstationname*.sth
SSL ca certificate	TWSca.crt
SSL random seed	*workstationname*.rnd

Configuring SSL attributes

Use the **composer** command line or the Dynamic Workload Console to update the workstation definition in the database. See the *IBM Workload Scheduler: User's Guide and Reference* for further information.

Configure the following attributes:

secureaddr
Defines the port used to listen for incoming SSL connections. This value must match the one defined in the **nm SSL port** local option of the workstation. It must be different from the **nm port** local option that defines the port used for normal communications. If **securitylevel** is specified but this attribute is missing, 31113 is used as the default value.

securitylevel
Specifies the type of SSL authentication for the workstation. It must have one of the following values:

enabled
The workstation uses SSL authentication only if its domain manager workstation or another fault-tolerant agent below it in the domain hierarchy requires it.

on
The workstation uses SSL authentication when it connects with its domain manager. The domain manager uses SSL authentication when it connects to its parent domain manager. The fault-tolerant agent refuses any incoming connection from its domain manager if it is not an SSL connection.

force
The workstation uses SSL authentication for all of its connections and accepts connections from both parent and subordinate domain managers. It will refuse any incoming connection if it is not an SSL connection.

If this attribute is omitted, the workstation is not configured for SSL connections. In this case, any value for **secureaddr** will be ignored. You should also set the **nm ssl port** local option to 0 to be sure that this port is not opened by netman. The following table describes the type of communication used for each type of **securitylevel** setting.

Table 70. Type of communication depending on the securitylevel value

Fault-tolerant agent (domain manager)	Domain manager (parent domain manager)	Connection type
-	-	TCP/IP
Enabled	-	TCP/IP
On	-	No connection
Force	-	No connection
-	On	TCP/IP
Enabled	On	TCP/IP
On	On	SSL
Force	On	SSL
-	Enabled	TCP/IP
Enabled	Enabled	TCP/IP
On	Enabled	SSL
Force	Enabled	SSL
-	Force	No connection

Table 70. Type of communication depending on the securitylevel value (continued)

Fault-tolerant agent (domain manager)	Domain manager (parent domain manager)	Connection type
Enabled	Force	SSL
On	Force	SSL
Force	Force	SSL

The following example shows a workstation definition that includes the security attributes:

```
cpuname MYWIN
os WNT
node apollo
tcpaddr 30112
secureaddr 32222
for maestro
autolink off
fullstatus on
securitylevel on
end
```

Configuring the SSL connection protocol for the network

About this task

To configure SSL for your network, perform the following steps:

1. Create an SSL directory under the *TWA_home*/TWS directory. By default, the path *TWA_home*/TWS/ssl is registered in the localopts file. If you create a directory with a name different from ssl in the *TWA_home*/TWS directory, then update the localopts file accordingly.
2. Copy openssl.cnf and openssl.exe to the SSL directory
3. Create as many private keys, certificates, and Trusted CA lists as you plan to use in your network.
4. For each workstation that will use SSL authentication:
   - Update its definition in the IBM Workload Scheduler database with the SSL attributes.
   - Add the SSL local options in the localopts file.

Although you are not required to follow a particular sequence, these tasks must all be completed to activate SSL support.

In IBM Workload Scheduler, SSL support is available for the fault-tolerant agents only (including the master domain manager and the domain managers), but not for the extended agents. If you want to use SSL authentication for a workstation that runs an extended agent, you must specify this parameter in the definition of the host workstation of the extended agent.

Configuring full SSL security

This section describes how to implement full SSL security when using an SSL connection for communication across the network by **netman** and **conman**. It contains the following topics:

Note: The full SSL security feature is not applicable to the communication between dynamic agents and the broker workstation that is defined for the master domain manager or the dynamic domain manager to which the dynamic agents are connected.

Overview: This feature provides the option to set a higher degree of SSL-based connection security on IBM Workload Scheduler networks in addition to the already available level of SSL security.

If you require a more complete degree of SSL protection, this feature supplies new configuration options to setup advanced connection security, otherwise you can use the standard settings documented above in this chapter.

The Full SSL security enhancements: Full SSL security support provides the following enhancements:

- TCP ports that can become security breaches are no longer left open.
- Traveling data, including communication headers and trailers, is now *totally* encrypted.

Compatibility between SSL support levels: The non-full and the full SSL support levels are mutually exclusive. That is, they cannot be configured simultaneously and cannot be enabled at the same time. If you enable full SSL support for a IBM Workload Scheduler network, any connection attempts by agents that are not configured for full SSL will be rejected by agents with full SSL support enabled. Vice versa, agents configured for full SSL support cannot communicate with the rest of a network set up for non-full SSL support.

Setting up full SSL security:
About this task

To set full SSL connection security for your network, you must, *in addition to all the steps described above in Chapter 7, "Setting connection security," on page 303)* configure the following options:

enSSLFullConnection (or sf)
Use `optman` on the master domain manager to set this global option to `Yes` to enable full SSL support for the network.

nm SSL full port
Edit the `localopts` file on every agent of the network (including the master domain manager) to set this local option to the port number used to listen for incoming SSL connections. Take note of the following:

- This port number is to be defined also for the `SECUREADDR` parameter in the workstation definition of the agent.
- In a full SSL security setup, the `nm SSL port` local option is to be set to zero.
- You must stop **`netman`** (**conman shut;wait**) and restart it (**StartUp**) after making the changes in `localopts`.
- Check that the `securitylevel` parameter in the workstation definition of each workstation using SSL is set at least to *enabled*.

Other than the changed value for `secureaddr`, no other changes are required in the workstation definitions to set up this feature.

Migrating a network to full SSL connection security:

About this task

Run the following steps to migrate your IBM Workload Scheduler version 8.3 production environment to full SSL connection security support. The scenario assumes that the network already runs on non-full SSL; that is, that the master and all the agents have:

- The `securitylevel` attribute set to `enabled`, `on`, or `force` in their workstation definition. On the master it is set to `enabled`.
- Either the `nm port` or the `nm SSL port` local option configured and the port number set as the value of the `secureaddr` attribute in their workstation definition.
- Group or individual private keys and certificates.

Proceed as follows:

1. Upgrade all the agents. The objective is to upgrade locally every agent in the network (including the master domain manager). You can perform this step over several days. On the master and on every agent:
   a. Install the fix containing the full SSL support feature.
   b. Add the `nm SSL full port` local option and set it to a port number.

   At this stage, the network is still operating on non-full SSL connection security.
2. Enable full SSL support in the network. Perform this step in one single time slot. To do this:
   a. Check that no firewall blocks the connection between the agents and their domain manager (and, optionally, the master domain manager).
   b. In the workstation definition of the master and of every agent set the value of the `secureaddr` attribute to the port number you configured for the `nm SSL full port` local option.
   c. Use Optman to set the `enSSLFullConnection` global option to `yes` in the database.
   d. Run `JnextPlan -for 0000` to make these settings operational.

   At this stage, the network is operating on full SSL connection security. Any agents left on SSL security can no longer communicate with the rest of the full SSL security network.

   The upgraded workstations still have the old SSL and TCP ports open in listening mode. The aim of the final step is to close them down.
3. Disable the old SSL and TCP ports on the master and on every agent. You can perform this step over several days. To do this, edit the local options file of every workstation as follows:
   - On the workstations that have the `securitylevel` attribute set to `enabled` or `on`, set the `nm SSL port` local option to 0.
   - On the workstations that have the `securitylevel` attribute set to `force`, set both `nm port` and `nm SSL port` local options to 0.

   At this stage, all the agents operate with the new SSL connections and all agents set on `securitylevel=force` listen only on the new SSL full port. From now on:
   - No bytes are sent in clear.
   - No active services are left in clear.
   - No TCP ports are left in listening mode on agents with `securitylevel=force`.

Configuring full SSL support for internetwork dependencies:

About this task

The network agent that resolves internetwork dependencies requires a particular setup for full SSL support.

To enable a network agent for full SSL support:

1. Configure both the hosting and the remote fault-tolerant agents for full SSL support.
2. On the hosting fault-tolerant agent copy or move the `netmth.opts` file from the *TWA_home*/TWS/config to the *TWA_home*/TWS/methods directories and add (and configure) the following options:

   SSL remote CPU
   The workstation name of the remote master or fault-tolerant agent.

   SSL remote full port
   The port number defined for full SSL support on the remote master or fault-tolerant agent.

   The local options that specify the private key and certificate on the hosting fault-tolerant agent
   These are documented in the "Setting local options" on page 34).

   Note that if the hosting fault-tolerant agent hosts more than one network agent, the *TWA_home*/TWS/methods directory contains one `netmth.opts` file for every defined network agent. In this case the complete name of each `netmth.opts` file must become:

   *network-agent-name*_netmth.opts

   If the *TWA_home*/TWS/methods directory contains both *network-agent-name*_netmth.opts and `netmth.opts` files, only *network-agent-name*_netmth.opts is used. If multiple agents are defined and the directory contains only `netmth.opts`, this file is used for all the network agents.

The following example adds full SSL support to the example described in *A sample network agent definition* in the IBM Workload Scheduler User's Guide and Reference:

- This is the workstation definition for the NETAGT network agent:

```
CPUNAME NETAGT
 DESCRIPTION "NETWORK AGENT"
 OS OTHER
 NODE MASTERA.ROME.TIVOLI.COM
 TCPADDR 31117
 FOR maestro
  HOST MASTERB
  ACCESS NETMTH
END
```

- These are the full SSL security options in the `netmeth.opts` file of NETAGT:

```
#####################################################
# Remote cpu parameters
#####################################################

SSL remote full port = 31119
SSL remote CPU = MASTERA

#####################################################
# Configuration Certificate
#####################################################

SSL key                ="C:\TWS\installations\SSL\XA.key"
SSL certificate        ="C:\TWS\installations\SSL\XA.crt"
```

```
SSL CA certificate      ="C:\TWS\installations\SSL\VeriSte.crt"
SSL key pwd             ="C:\TWS\installations\SSL\XA.sth"
SSL certificate chain   ="C:\TWS\installations\SSL\TWSCertificateChain.crt"
SSL random seed         ="C:\TWS\installations\SSL\random_file.rnd"
SSL auth mode           =cpu
SSL auth string        =tws
```

Note: The SSL configuration certificate options must refer to the private key and certificate defined on the hosting fault-tolerant agent.

- This is the workstation definition for MASTERA (the remote workstation):

```
CPUNAME MASTERA
  OS WNT
  NODE 9.168.68.55 TCPADDR 31117
  SECUREADDR 31119
  DOMAIN NTWKA
  FOR MAESTRO
    TYPE MANAGER
    AUTOLINK ON
    BEHINDFIREWALL OFF
    SECURITYLEVEL enabled
    FULLSTATUS ON
    SERVER H
END
```

Scenario: HTTPS for the command-line clients

You can have one of the following scenarios:

- SSL connection between the command-line utilities (**composer** and **conman**) on the master domain manager and the connector installed in the master domain manager.
- SSL connection between the remote command-line client installed on a workstation and the remote master domain manager workstation.

SSL connection by using default certificates

For more information about the SSL default connection, see "Configuring SSL using the predefined certificate" on page 337.

SSL connection by using your certificates

For more information about how to create and enable your SSL certificates, see "Using a customized certificate" on page 338.

Customizing the SSL connection for a command-line client

About this task

IBM Workload Scheduler command-line clients connect to the connector through HTTP or HTTPS. The default connection type is HTTPS. If the command-line connects through a proxy, use the HTTP connection protocol as HTTPS is not supported for this type of configuration.

You configure the connection protocol as described in "Configuring command-line client access authentication" on page 93. If you have previously not been using SSL for the command line client access, you will need to change at least the following parameters:

proxy Specify the IP address or the server name for the proxy server.

proxy port
Specify the listening port of the proxy server.

protocol
Specify the protocol type as HTTP or HTTPS.

port Specify the port required by the protocol you set in the **protocol** option. The default is 31115 for HTTP and 31116 for HTTPS.

The HTTPS connection protocol offers the following additional security features:

- Data encryption between the command-line utility and the connector
- Optional server authentication by validating the server certificates

You can activate optional server authentication in one of the following ways:

- "Configuring SSL using the predefined certificate"
- "Configuring multiple SSL communication instances"
- "Using a customized certificate" on page 338

Configuring SSL using the predefined certificate

About this task

To customize the SSL connection for the command-line client using the predefined certificate, perform the following steps:

1. Stop the WebSphere Application Server using the **conman stopappserver** command.
2. Extract the certificate from the `TWA_home/WAS/TWSprofile/etc/TWSServerKeyFile.jks` keystore:

```
TWA_home/WAS//java/jre/bin/keytool -export
         -alias server
         -rfc
        -file server.crt
         -keystore TWA_home/WAS/TWSprofile/etc/TWSServerKeyFile.jks
        -storepass default
```

3. Copy the .crt certificate (`server.crt` in the previous example) to each workstation that has a command-line client installed, copying it to the path set in the **cli ssl server certificate** command-line client option (see next step).
4. Set the **cli ssl server auth** and **cli ssl server certificate** command-line client options in the `localopts` file. See "Setting local options" on page 34 for details about how to set these options.
5. Start the WebSphere Application Server using the **conman startappserver** command.

Configuring multiple SSL communication instances

About this task

To customize the SSL connection for the command-line client for multiple connections to WebSphere Application Server, perform the following steps:

1. Stop WebSphere Application Server using the **conman stopappserver** command.
2. Extract a certificate from `TWSServerKeyFile.jks` keystore for each instance.

```
keytool -export
        -alias tws
        -rfc
        -file server.crt
        -keystore ServerKeyFile.jks
        -storepass default
```

.

3. Extract the hash number for each exported certificate:

```
openssl x509
        -hash
        -noout
        -in keyname
```

4. Rename each certificate file with the exported key.
5. Copy the renamed certificates to each workstation that has a command-line client installed.
6. Set the **cli ssl server auth** and **cli ssl trusted dir** command-line client options in the `localopts` file. See "Setting local options" on page 34.
7. Start the WebSphere Application Server using the **conman startappserver** command.

Using a customized certificate

About this task

To customize the SSL certificate and keystore, perform the following steps:

1. Create a new RSA and extract the key for the server keystore TWSServerKeyFile.jks.
2. Import the certificate in PEM format:

```
keytool -import
        -alias tws
        -file server.crt
        -trustcacerts
        -noprompt
        -keystore TWSClientTrustFile.jks
        -storepass default
```

3. Follow the steps in "Configuring SSL using the predefined certificate" on page 337.

Note: When you want to customize certificates for multiple instances, perform these steps for each instance.

Using SSL for event-driven workload automation (EDWA) behind firewalls

This feature allows a domain manager to be run as a reverse proxy for HyperText Transfer Protocol (HTTP) and Event Integration Facility (EIF) protocols, forwarding traffic to the Event Processor. An option, enabled using the **optman** command-line program, allows you to choose if workstations that are behind a firewall must connect to the domain manager instead of to the event processor, causing the new proxy on the domain manager to forward its traffic to the event processor.

Restriction: This configuration is not supported if the agent workstation is a dynamic agent.

The incoming traffic is rerouted as follows:

- If an agent is behind a firewall, the traffic is routed to the domain manager on the agent. If an agent is not behind a firewall, the traffic is sent directly to the event processor.
- If domain managers have child nodes behind a firewall, the traffic is rerouted to the event processor.
- Primary domain managers always reroute traffic to the current event processor.
- Lower level domain managers reroute traffic to upper level domain managers if they are behind a firewall, or to the event processor if they are not behind a firewall.

To use this feature, perform the following steps:

1. Enable the feature by setting the **optman** option to `yes`. The default value is `no`:
   `enEventDrivenWorkloadAutomationProxy | pr = {yes|no}`
2. In the workstation definition in the database for the agent, set the **`behindfirewall`** attribute to ON.
3. Configure OpenSSL or GSKit on the domain manager.

For details on how to set the attribute **`behindfirewall`**, see the *IBM Workload Scheduler: User's Guide and Reference*.

Configuring IBM Workload Scheduler to use LDAP

About this task

To use LDAP configured in SSL with IBM Workload Scheduler, perform the following steps:

Procedure

1. Import the LDAP public key in the truststore of the IBM Workload Scheduler server, by storing it in the `TWSServerTrustFile.jks` file located in *<WAS_profile_path>*`/etc`, where the default value of *<WAS_profile_path>* is *<TWA_home>*`/WAS/TWSprofile`.
2. Import the LDAP public key in the truststore of the IBM Workload Scheduler client, by storing it in the `TWSClientTrustFile.jks` file located in *<WAS_profile_path>*`/etc`.

FIPS compliance

This section describes Federal Information Processing Standards (FIPS) compliance. It is divided into the following topics:

FIPS overview

Federal Information Processing Standards (FIPS) are standards and guidelines issued by the National Institute of Standards and Technology (NIST) for federal government computer systems. FIPS are developed when there are compelling federal government requirements for standards, such as for security and interoperability, but acceptable industry standards or solutions do not exist. Government agencies and financial institutions use these standards to ensure that the products conform to specified security requirements.

IBM Workload Automation uses cryptographic modules that are compliant with the Federal Information Processing Standard FIPS-140-2. Certificates used internally are encrypted using FIPS-approved cryptography algorithms. FIPS-approved modules can optionally be used for the transmission of data.

To satisfy the FIPS 140-2 requirement, you must use IBM Global Security Kit (GSKit) version 7d run time dynamic libraries instead of OpenSSL. GSKit uses IBM Crypto for C version 1.4.5 which is FIPS 140-2 level 1 certified by the certificate number 755. See http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/1401val2007.htm. IBM Java JSSE FIPS 140-2 Cryptographic is another module used by IBM Workload Automation. It has the certificate number 409.

If you are currently using SSL for secure connections across the network, to ensure FIPS compliance, you must use GSKit for secure connections instead of OpenSSL Toolkit. GSKit is automatically installed with IBM Workload Scheduler. It is based on dynamic libraries and offers several utilities for certificate management.

To comply with FIPS, all components of IBM Workload Automation must be FIPS-compliant. You must use Dynamic Workload Console or the IBM Workload Scheduler command line as the interface to IBM Workload Scheduler. Additionally, you must use DB2 as your IBM Workload Scheduler database.

If FIPS compliance is not of concern to your organization, you can continue to use SSL for secure connections across your network.

Components of IBM Workload Automation not FIPS-compliant cannot communicate with components of IBM Workload Automation FIPS-compliant.

To set FIPS compliance for your network, perform the procedures described in the following sections:

- To create FIPS certificates, see "Using FIPS certificates."
- To configure SSL for FIPS-compliance, see "Configuring SSL to be FIPS-compliant" on page 344.
- To configure your DB2 database for FIPS-compliance, see "Configuring DB2 for FIPS" on page 347.

Using FIPS certificates

About this task

To ensure your network is FIPS-compliant, create FIPS certificates as follows:

- If you do not already have SSL certificates, see "Using fresh FIPS certificates" on page 341.
- If you already have SSL certificates but are switching to GSKit, see "Switching from OpenSSL to GSKit" on page 342.

If you are using FIPS certificates, you must use SSL parameters for communication over the network. During the installation or upgrade to IBM Workload Scheduler version 8.5.1, note that default SSL certificates are located in the following directories:

TWS_InstallDir\TWS\ssl\GSKit
TWS_InstallDir\TWS\ssl\OpenSSL

Using fresh FIPS certificates

Create FIPS certificates for communication between workstations by using the –fips option in the GSKit command line utility. You can create FIPS certificates in the following ways:

- Use the default FIPS certificates existing on each IBM Workload Scheduler agent in the network. Note that the default FIPS certificates are not secure.
- Create your own secure FIPS certificates. See "Creating your own FIPS certificates."

Creating your own FIPS certificates: Use the **gsk7capicmd** command line utility to:

- Create your own Certificate Authority (CA).
- Create a self-signed CA certificate (x.509 structure) for your CA.
- Export the CA certificate in PEM format.

Creating your own Certificate Authority: Create the CA on any workstation in your network. Run the following steps only once to create a CA that will be used each time a new certificate needs to be created and signed.

1. Enter the following command to create the CMS key database "ca.kdb" with password "password00" that expires after 1000 days.

   ```
   gsk7capicmd -keydb -create -db ca.kdb -pw password00 -stash -expire 1000 -fips
   ```

2. Enter the following command to create the self-signed certificate with label "CA certificate" using the distinguish name "CN=CA certificate,O=IBM,OU=TWS,C=IT". The certificate expires after 1000 days.

   ```
   gsk7capicmd -cert -create  -db ca.kdb -pw password00 -label "CA certificate"
        -size 2048 -expire 1000 -dn "CN=CA certificate,O=IBM,OU=TWS,C=IT"
   ```

3. Enter the following command to extract the CA certificate into external file "ca.crt". The certificate is addressed by the corresponding label.

   ```
   gsk7capicmd -cert -extract -db ca.kdb -pw password00 -label "CA certificate"
         -target CA.crt
   ```

This file will contain the public certificate of the certificate authority.

Creating a certificate for the IBM Workload Scheduler agent: Perform the following steps to create certificates that are signed by a local common trusted CA on every IBM Workload Scheduler agent in your network.

1. Enter the following command to create a default CMS key database client.kdb" with password "password02" that expires after 1000 days. The password is also stored in stash file "client.sth".

   ```
   gsk7capicmd -keydb -create -db client.kdb -pw password02
       -stash -expire 1000 -fips
   ```

2. Enter the following command to add the CA certificate as trusted in the CMS key database. The label "CA certificate client" is used to address that certificate.

   ```
   gsk7capicmd -cert -add -db client.kdb -pw password02
       -label "CA certificate client" -trust enable -file CA.crt
       -format ascii -fips
   ```

3. Enter the following command to create the client certificate request based on 2048 bits key, with label **"Client TWS85 Certificate"** and distinguish name **"CN=Client TWS85,O=IBM,OU=TWS,C=IT"**. The certificate request "client.csr" is generated and the private key is created in the key database client.kdb.

   ```
   gsk7capicmd -certreq -create -db client.kdb -pw password02
       -label "Client TWS85 Certificate" -size 2048 -file client.csr
       –dn "CN=Client  TWS85,O=IBM,OU=TWS,C=IT" -fips
   ```

4. Enter the following command so that the CA signs the client's certificate request and generates a new signed in file "client.crt".

   ```
   gsk7capicmd -cert -sign -db ca.kdb -pw password00 -label "CA certificate"
        -target client.crt -expire 365 -file client.csr -fips
   ```

5. Enter the following command to import the signed certificate "client.crt" in the CMS key database "client.kdb".

   ```
   gsk7capicmd -cert -receive -db client.kdb -pw password02 -file client.crt -fips
   ```

You can repeat these steps above for all agents or you can use the same certificate for all agents, depending on your security policies and IBM Workload Scheduler localopts configurations.

Switching from OpenSSL to GSKit

About this task

This section describes how to migrate your OpenSSL certificates to GSKit certificates.

The following is a list of certificate formats that can be migrated to the GSKit format, **KDB**:

- **PEM**: Used by OpenSSL
- **JKS**: Used by Java and WebSphere Application Server
- **PKCS12**: Used by Microsoft applications and Internet Explorer

To migrate certificates, you may use one or more of the following tools:

- **gsk8capicmd**: Native command line provided by GSKit
- **openssl**: Native command line provided by OpenSSL
- **ikeyman**: Optional graphical interface provided by GSKit
- **keytool**: Optional graphical interface provided by Java Virtual Machine (JVM)

Note: Be sure to backup your original certificates before migrating them to GSKit format.

To migrate your certificates, perform the following steps:

1. "Configuring the tool environment"
2. "Migrating the certificates" on page 343

Configuring the tool environment: This section describes the commands you must run to configure gsk8capicmd and openssl.

Configuring gsk8capicmd:

gsk8capicmd on 32 bit

```
set PATH=C:\Program Files\IBM\TWA\TWS\GSkit32\8\lib; C:\Program
Files\IBM\TWA\TWS\GSkit32\8\bin;%PATH%
```

gsk8capicmd_64 on 64 bit

```
set PATH=C:\Program Files\IBM\TWA\TWS\GSkit64\8\lib64; C:\Program
Files\IBM\TWA\TWS\GSkit64\8\bin;%PATH%
```

Configuring openssl:

UNIX `tws_env.sh`

Windows

`tws_env.cmd`

Migrating the certificates: This section describes the commands you must run to migrate certificates to the FIPS-compliant format, KDB.

Note that PEM format cannot be directly converted to KDB format; you must first convert PEM to PKCS12 and then to KDB.

The following list describes the command you must run to convert from one format to another:

JKS format to KDB format

gsk7cmd -keydb -convert -db TWSClientKeyFile.jks -pw default -old_format jks -new_format cms

gsk7cmd -keydb -convert -db TWSClientTrustFile.kdb -pw default -old_format cms -new_format jks

PKCS12 format to KDB format

gsk7capicmd -cert -export -target TWSClientKeyFile_new.kdb -db TWSClientKeyFileP12.P12 -fips -target_type cms -type pkcs12

PKCS12 format to PEM format

openssl pkcs12 -in TWSClientKeyFileP12.P12 -out TWSClientKeyFile.pem

PEM format to PKCS12 format

openssl pkcs12 -export -in TWSClientKeyFile.pem -out cred.p12

KDB format to PKCS12 format

gsk7capicmd -cert -export -db TWSClientKeyFile.kdb -target TWSClientKeyFileP12.P12 -fips -target_type pkcs12 -type cms

Converting PEM certificates to CMS certificates: This section describes the procedure to convert PEM (OpenSSL) certificates to CMS (GSKit) certificates. The examples in this section use the following input and output files.

Input files

Personal certificate file: *CPU1.crt*
Personal key of certificate file: *CPU1.key*
Certificate of CA file: *TWSca.crt*
Stash file: *CPU1.sth*

Output files

Keystore database file: *TWS.kdb*
Stash file: *TWS.sth*
Label of your certificate: *CPU1*

To migrate OpenSSL certificates to GSKit certificates, perform the following procedure:

1. Merge the public and private keys in a new temporary file called **all.pem** by running the following commands:

   UNIX cat CPU2.crt CPU2.key > all.pem

   Windows
   type CPU1.crt CPU1.key > all.pem
2. If you do not already know the password, extract it from the stash file by running `openssl base64 -d -in CPU1.sth`.
3. Choose a password for the new keystore database. You can reuse the old password.
4. Choose a label for your personal certificate and personal key (in this example, CPU1) and create the PKCS12 database that contains the labels. You use the name, CPU1, as the label of the new keystore database. To create the PKCS12 database, run the following:

   ```
   openssl pkcs12 -export -in all.pem -out TWS.p12 -name CPU1  -passin pass:
           password1 -passout pass:password2
   ```

   where *password1* is the password extracted from the stash file and *password2* is is the new password to manage the new keystore database.
5. Convert the PKCS12 database from TWS.p12 to the CMS database, TWS.kdb by running the following:

   ```
   gsk7capicmd -cert -import -target TWS.kdb -db TWS.p12 -target_type cms
            -type pkcs12 -label CPU1 -target_pw "password2" -pw "password3"
   ```

   where *password2* is the old password that you extracted from the stash file, CPU1.sth and *password3* is the new password.
6. Choose a label for your Certification Authority contained in TWSca.crt. For this example, it is *TWSca*.
7. Add the certificate of the Certification Authority into your TWS.kdb file by running:

   ```
   gsk7capicmd -cert -add -db TWS.kdb -label TWSca -trust -file TWSca.crt
           -format ascii -pw "password"
   ```
8. Delete all .pem files.

Configuring SSL to be FIPS-compliant

About this task

To configure SSL to be FIPS-compliant, perform the following procedures:

- Set localopts parameters. See "Setting localopts parameters for FIPS" on page 345.
- Configure WebSphere Application Server. See "Configuring WebSphere Application Server for FIPS" on page 345.
- Configure the Tivoli event integration facility port. See "Configuring the Tivoli event integration facility port" on page 347.

Note:

If you are using dynamic workload broker for dynamic scheduling in your network, note that the workstation of type **BROKER** does not support SSL. All IBM Workload Scheduler workstations must communicate with the workstation of type **BROKER** using TCP/IP protocol.

Setting localopts parameters for FIPS
About this task

To set your environment for FIPS, set the following local option on every IBM Workload Scheduler agent in the network.

SSL Fips enabled = *yes*

The following example applies to a Windows agent. Set the following local options for the engine:

SSL keystore file = *"<TWA_home>\TWS\ssl\GSKit\TWSClientKeyStore.kdb"*
SSL certificate keystore label = *"client"*
SSL keystore pwd = *"<TWA_home>\TWS\ssl\GSKit\TWSClientKeyStore.sth"*

where <*TWA_home*> is the installation directory of the instance of IBM Workload Automation where the agent is installed.

Set the following local options for the CLI:

CLI SSL keystore file =
"<TWA_home>\TWS\ssl\GSKit\TWSClientKeyStore.kdb"
CLI SSL certificate keystore label = *"client"*
CLI SSL keystore pwd =
"<TWA_home>\TWS\ssl\GSKit\TWSClientKeyStore.sth"

where <*TWA_home*> is the installation directory of the instance of IBM Workload Automation where the agent is installed.

For more information about setting local options and the **localopts** file, see "Setting local options" on page 34.

Note: On Windows workstations, the user, **SYSTEM**, must have read-permissions to read the GSKit FIPS certificates.

Configuring WebSphere Application Server for FIPS
About this task

To be FIPS-compliant, you must configure WebSphere Application Server for IBM Workload Scheduler.

The section describes how to:

- Configure WebSphere Application Server for IBM Workload Scheduler. See "Configuring WebSphere Application Server for IBM Workload Scheduler."
- Configure the Tivoli event integration facility port. See "Configuring the Tivoli event integration facility port" on page 347.

Configuring WebSphere Application Server for IBM Workload Scheduler: To configure WebSphere Application Server for FIPS compliance, perform the following steps:

1. In the WebSphere Application Server administration interface, click **Security > SSL certificate and key management**. Select **Use the United States Federal Information Processing Standard (FIPS) algorithms** and click **Apply**. Alternatively, you can use wastools, running **changeSecurityProperties** to change the following parameter:

```
useFIPS=true
```

2. In the **profile_root/properties/ssl.client.props** file, set the following parameters:
   - **com.ibm.security.useFIPS**=*true*
   - **com.ibm.ssl.protocol**=*SSL_TLS*
3. If you have an administrative client that uses a SOAP connector, add the following line to the **profile_root/properties/soap.client.props** file:
   ```
   com.ibm.ssl.contextProvider=IBMJSSEFIPS
   ```
4. Edit the SDK java.security file located in the WASHOME/java_1.8_64/jre/lib/security directory to insert the **IBMJCEFIPS** provider (**com.ibm.crypto.fips.provider.IBMJCEFIPS**). **IBMJCEFIPS** must precede the **IBMJCE** provider in the provider list.

   The following is an example of the edited SDK java.security file:
   ```
   security.provider.1=com.ibm.crypto.fips.provider.IBMJCEFIPS
   security.provider.2=com.ibm.crypto.provider.IBMJCE
   security.provider.3=com.ibm.jsse.IBMJSSEProvider
   security.provider.4=com.ibm.jsse2.IBMJSSEProvider2
   security.provider.5=com.ibm.security.jgss.IBMJGSSProvider
   security.provider.6=com.ibm.security.cert.IBMCertPath
   security.provider.7=com.ibm.crypto.pkcs11.provider.IBMPKCS11
   security.provider.8=com.ibm.security.cmskeystore.CMSProvider
   security.provider.9=com.ibm.security.jgss.mech.spnego.IBMSPNEGO
   ```

   The following is an example of the edited java.security file if you are using the Oracle Java SE Development Kit:
   ```
   security.provider.1=sun.security.provider.Sun
   security.provider.2=com.ibm.crypto.fips.provider.IBMJCEFIPS
   security.provider.3=com.ibm.crypto.provider.IBMJCE
   security.provider.4=com.ibm.jsse.IBMJSSEProvider
   security.provider.5=com.ibm.jsse2.IBMJSSEProvider2
   security.provider.6=com.ibm.security.jgss.IBMJGSSProvider
   security.provider.7=com.ibm.security.cert.IBMCertPath
   security.provider.8=com.ibm.i5os.jsse.JSSEProvider
   #security.provider.8=com.ibm.crypto.pkcs11.provider.IBMPKCS11
   security.provider.9=com.ibm.security.jgss.mech.spnego.IBMSPNEGO
   ```
5. Restart the WebSphere Application Server.

Note: For additional information about WebSphere Application Server and FIPS, see the WebSphere Application Server documentation.

Unconfiguring the FIPS provider: To unconfigure the FIPS provider, reverse the changes that you made in "Configuring WebSphere Application Server for FIPS" on page 345. After you reverse the changes, verify that you have made the following changes to the ssl.client.props, soap.client.props, and java.security files:

- In the ssl.client.props file, change the com.ibm.security.useFIPS value to false.
- In the java.security file, change the FIPS provider to a non-FIPS provider.
- If you are using the SDK java.security file, change the first provider to a non-FIPS provider as shown in the following example:
  ```
  #security.provider.1=com.ibm.crypto.fips.provider.IBMJCEFIPS
  security.provider.1=com.ibm.crypto.provider.IBMJCE
  security.provider.2=com.ibm.jsse.IBMJSSEProvider
  security.provider.3=com.ibm.jsse2.IBMJSSEProvider2
  security.provider.4=com.ibm.security.jgss.IBMJGSSProvider
  security.provider.5=com.ibm.security.cert.IBMCertPath
  #security.provider.6=com.ibm.crypto.pkcs11.provider.IBMPKCS11
  ```
- If you are using the Oracle JDK java.security file, change the third provider to a non-FIPS provider as shown in the following example:

```
security.provider.1=sun.security.provider.Sun
security.provider.2=com.ibm.security.jgss.IBMJGSSProvider
#security.provider.3=com.ibm.crypto.fips.provider.IBMJCEFIPS
security.provider.3=com.ibm.crypto.provider.IBMJCE
security.provider.4=com.ibm.jsse.IBMJSSEProvider
security.provider.5=com.ibm.jsse2.IBMJSSEProvider2
security.provider.6=com.ibm.security.cert.IBMCertPath
#security.provider.7=com.ibm.crypto.pkcs11.provider.IBMPKCS11
```

- This step applies only if you added the default JSSE socket factories parameters to the SDK java.security file as described in "Configuring DB2 for FIPS." If you added them, remove the following parameters:

```
ssl.SocketFactory.provider=com.ibm.jsse2.SSLSocketFactoryImpl
ssl.ServerSocketFactory.provider=com.ibm.jsse2.SSLServerSocketFactoryImpl
```

Configuring the Tivoli event integration facility port

About this task

The Tivoli event integration facility port for SSL, **eventProcessorEIFSSLPort**, is used in event management. For the Tivoli event integration facility port to communicate in FIPS mode, you must first configure WebSphere Application Server for FIPS. See "Configuring WebSphere Application Server for IBM Workload Scheduler" on page 345.

To configure the Tivoli event integration facility port for SSL, perform the following steps:

1. Ensure that you have set the **SSL Fips Enabled** local option on every agent, as described in "Setting localopts parameters for FIPS" on page 345.
2. Set the global option for the port by using optman, as follows:

   ```
   eventProcessorEIFSSLPort / ef = portnumber
   ```

   where *portnumber* is the number of any free port on your network.
3. To update the Symphony file, run **JnextPlan -for 0000**.
4. Restart the EventProcessor by using the **conman stopevtp** and **conman startevtp** commands.
5. Restart the IBM Workload Scheduler monitoring engine with the conman commands, **stopmon** and **startmon**.

Configuring DB2 for FIPS

To configure DB2 for FIPS compliance, perform the following procedures on the supported DB2 version you are using:

- "Configuring DB2"
- "Configuring the DB2 connection to IBM Workload Scheduler" on page 350

Note: If you want to create your own DB2 certificates, see the DB2 documentation.

Configuring DB2

About this task

To configure supported versions of DB2 for FIPS compliance, perform the following procedure:

Note: For information about supported versions of DB2, refer to the System Requirements Document at http://www-01.ibm.com/support/docview.wss?rs=672&uid=swg27048858.

1. Ensure that the path to the GSKit libraries is included in the relevant environment variables. The names of the environment variables vary depending on the operating system, as follows:

   UNIX and Linux
   LIBPATH, LD_LIBRARY_PATH, or SHLIB_PATH environment variables. Specify this information in the .profile file for the DB2 instance owner.

   Windows
   PATH environment variable. For example, `c:\Program Files\IBM\gsk8\lib`.

   GSKit is automatically included when you install the DB2 database system.

   On Windows 32-bit operating systems
   the GSKit libraries are located in C:\Program Files\IBM\GSK8\lib. In this case, the system PATH must include C:\Program Files\IBM\GSK8\lib.

   On Windows 64-bit operating systems
   the 64-bit GSKit libraries are located in C:\Program Files\IBM\GSK8\lib64 and the 32-bit GSKit libraries are located in C:\Program Files (x86)\IBM\GSK8\lib.

   On UNIX and Linux operating systems,
   the GSKit libraries are located in sqllib/lib. Therefore, the LIBPATH, SHLIB_PATH or LD_LIBRARY_PATH environment variables must include sqllib/lib.

   On non-Windows operating systems
   the DB2 database manager installs GSKit locally, and for a given instance, the GSKit libraries might be located in sqllib/lib or sqllib/lib64.

2. To set up your DB2 server for SSL support, log in as the DB2 instance owner and set the following configuration parameters and the **DB2COMM** registry variable. Use the **db2 update dbm cfg using** ***parameter_name parameter_value*** command, where

   parameter_name
   Is the name of the parameter to be set.

   parameter_value
   Is the value of the parameter to be set.

   a. Set the **ssl_svr_keydb** configuration parameter to the fully qualified path of the key database file. For example:

      ```
      C:\TWS\installations\tws850cli\TWS\ssl\gskit\TWSClientKeyStore.kdb
      ```

      where:

      TWSClientKeyStore.kdb
      Is the fully-qualified file name of the KeyStore that stores the DB2 certificate, and the trusted certificates, for example the certificates for the WebSphere Application Server to connect to. This KeyStore can be the same one you specified in the localopts parameters. See "Setting localopts parameters for FIPS" on page 345. Note that it must be recognized by the JKS WebSphere Application Server certificate.

      If **ssl_svr_keydb** is null (unset), SSL support is not enabled.

b. Set the **ssl_svr_stash** configuration parameter to the fully qualified path of the stash file. For example:

```
C:\TWS\installations\tws850cli\TWS\ssl\gskit\TWSClientKeyStore.sth
```

If **ssl_svr_stash** is null (unset), SSL support is not enabled.

c. Set the **ssl_svr_label** configuration parameter to the label of the digital certificate of the server. If **ssl_svr_label** is not set, the default certificate in the key database is used. If there is no default certificate in the key database, SSL is not enabled. For example:

```
"client"
```

d. Set the **ssl_svcename** configuration parameter to the port that the DB2 database system listens on for SSL connections. If TCP/IP and SSL are both enabled (the **DB2COMM** registry variable is set to 'TCPIP, SSL'), set **ssl_svcename** to a different port than the port to which **svcename** is set. The **svcename** configuration parameter sets the port that the DB2 database system listens on for TCP/IP connections. If you set **ssl_svcename** to the same port as **svcename**, neither TCP/IP or SSL are enabled. If **ssl_svcename** is null (unset), SSL support is not enabled.

Note:

1) In HADR environments, do not set **hadr_local_svc** on the primary or standby database system to the same value as you set for **ssl_svcename**. Also, do not set **hadr_local_svc** to the same value as **svcename**, or **svcename** plus one.
2) When the **DB2COMM** registry variable is set to 'TCPIP,SSL', if TCPIP support is not properly enabled, for example due to the **svcename** configuration parameter being set to null, the error SQL5043N is returned and SSL support is not enabled.

e. (Optional) If you want to specify which cipher suites the server can use, set the **ssl_cipherspecs** configuration parameter. If you leave **ssl_cipherspecs** as null (unset), this allows GSKit to pick the strongest available cipher suite that is supported by both the client and the server.

f. Add the value SSL to the **DB2COMM** registry variable. For example:

```
db2set -i db2inst1 DB2COMM=SSL
```

The database manager can support multiple protocols at the same time. For example, to enable both TCP/IP and SSL communication protocols:

```
db2set -i db2inst1 DB2COMM=SSL,TCPIP
```

where:

db2inst1
Is the DB2 instance name.

Note: During the installation of a IBM Workload Scheduler master domain manager or backup master domain manager, it is necessary to enable the DB2 TCPIP port. DB2 can support both TCP/IP and SSL communications protocols at the same time. The DB2 administrator can set the TCPIP port with the command **db2set DB2COMM=TCPIP, SSL**. Use this command if you are installing a IBM Workload Scheduler master domain manager or backup master domain manager and already have a FIPS-enabled instance of DB2. After installation, you can choose to reset DB2COMM with only SSL.

g. Restart the DB2 instance. For example:

```
db2stop
db2starts
```

3. Insert the following default JSSE socket factories parameters in the java.security file of the IBM Workload Scheduler WebSphere Application Server:

   ```
   ssl.SocketFactory.provider=com.ibm.jsse2.SSLSocketFactoryImpl
   ssl.ServerSocketFactory.provider=com.ibm.jsse2.SSLServerSocketFactoryImpl
   ```

4. Restart DB2.

Note: It is not necessary to update the DB2 JVM. This is because you already updated the JVM of WebSphere Application Server in the procedure described in "Configuring WebSphere Application Server for FIPS" on page 345.

For more information about how to configure DB2 to be FIPS compliant, see the DB2 documentation that describes how to configure Secure Sockets Layer (SSL) support in a DB2 instance.

Configuring the DB2 connection to IBM Workload Scheduler

About this task

After configuring DB2, you must configure IBM Workload Scheduler to communicate with the new settings of DB2. Perform the following procedure:

1. Modify the DB2 DataSource properties in wastools by running **showDataSourceProperties** and **changeDataSourceProperties** to include the following parameters:

   ```
   DB2Type4PortNumber=nnnnn
   DB2Type4SslConnection=true
   ```

   where *nnnnn* is the SSL DB2 port number.
2. Restart the WebSphere Application Server.

Using Dynamic Workload Console and FIPS

About this task

To ensure that you connect to Dynamic Workload Console using FIPS, perform the following steps:

1. Enable Transport Layer Security (TLS) in your browser as follows:
   - To enable TLS in Internet Explorer, open the browser and click **Tools > Internet Options**. On the Advanced tab, select **Use TLS 1.0**.
   - To enable TLS in Mozilla Firefox, open the browser and click **Tools >Options >Advanced**. On the Encryption tab, select **Use TLS 1.0**.
   - To enable TLS on other internet browsers, see the product documentation for that browser.
2. Depending on your configuration, perform one of the following procedures:
   - Dynamic Workload Console on a WebSphere Application Server:
     - Ensure that the WebSphere Application Server is FIPS-compliant. See "Configuring WebSphere Application Server for FIPS" on page 345.
   - Dynamic Workload Console with a DB2 settings repository:
     - Ensure that DB2 is FIPS-compliant. See "Configuring DB2 for FIPS" on page 347.
     - Ensure that DB2 and Dynamic Workload Console are mutually trusted by exchanging their certificates on their truststore. Using default certificates:
       - Keystore for Dynamic Workload Console:

```
C:\Program Files\IBM\JazzSM\profile\config\cells\JazzSMNode01Cell\nodes\
  JazzSMNode01\key.p12
```

- Truststore for Dynamic Workload Console:

```
C:\Program Files\IBM\JazzSM\profile\config\cells\JazzSMNode01Cell\nodes\
  JazzSMNode01\trust.p12
```

- ssl_svr_keydb (path of the key database file) configured for Dynamic Workload Console in FIPS:

```
C:\Program Files\IBM\TWA\TWS\ssl\GSKit\TWSClientKeyStore.kdb.
```

- Extract Dynamic Workload Console certificate:

```
C:\Program Files (x86)\IBM\WebSphere\AppServer\java_1.8_64\jre\
bin\ikeycmd
 -cert -extract -db <DWC keystore> -label default -target
c:\temp\tdwc.arm
 -pw WebAS
```

- Extract DB2 certificate:

```
C:\Program Files (x86)\IBM\WebSphere\AppServer\java_1.8_64\jre\bin
\ikeycmd
 -cert -extract -db <ssl_svr_keydb> -label client -target
c:\temp\db2.arm
 -pw default
```

- Import Dynamic Workload Console certificate into DB2 ssl_svr_keydb

```
C:\Program Files (x86)\IBM\WebSphere\AppServer\java_1.8_64\jre\
bin\ikeycmd
 -cert -add -db <ssl_svr_keydb> -file "c:\temp\tdwc.arm"
 -label tdwc
 -pw default -type cms -trust enable
```

- Import DB2 certificate into Dynamic Workload Console truststore

```
C:\Program Files (x86)\IBM\WebSphere\AppServer\java_1.8_64\jre\
bin\ikeycmd
 -cert -add -db <ssl_svr_keydb> -file "c:\temp\db2.arm"
-label db2 -pw WebAS
 -type pkcs12 -trust enable
```

3.

a. To ensure the required SSL connection between DB2 and Dynamic Workload Console, perform the following procedure:

1) Go to the IBM Workload Scheduler wastools directory and modify the TDWCDataSource properties to include the following parameters:

```
useSslConnection=true
deleteAndRecreate=true
databasePort=nnnnn
```

where *nnnnn* is the SSL DB2 port number.

2) Run **installTDWCDataSource** by entering the following commands:

UNIX and Linux operating systems
InstallTDWCDataSource.sh TDWCDataSource.properties

Windows operating systems
InstallTDWCDataSource.bat TDWCDataSource.properties

b. Restart the WebSphere Application Server and DB2.

4. If you are using Dynamic workload broker, set a secure connection by performing the following:

a. In Dynamic Workload Console, access Dynamic Workload Broker and expand the **Configuration** menu.

b. Click **Server Connections**.

c. In the Server Connections screen, select **Use Secure Connection**.
d. Click **OK**.

5. If you want to configure an SSL connection with a IBM Workload Scheduler for z/OS engine, launch one of the following utilities, setting the `useSSL` parameter to **true**:

   createZosEngine
   use this utility if you have not created a connection with IBM Workload Scheduler for z/OS engine, yet.
   - On Windows operating systems, launch: `\wastools\createZosEngine.bat`
   - On UNIX and Linux operating systems, launch: `/wastools/createZosEngine.sh`

   updateZosEngine
   use this utility if you have already created a connection with IBM Workload Scheduler for z/OS engine and you want to update its configuration.
   - On Windows operating systems, launch: `\wastools\updateZosEngine.bat`
   - On UNIX and Linux operating systems, launch: `/wastools/updateZosEngine.sh`

Note: To enable communication between Dynamic Workload Console and DB2, configure the Java system properties in Dynamic Workload Console to use the trustStore. To do this, set the following Java system properties:

```
javax.net.ssl.trustStore
javax.net.ssl.trustStorePassword
```

For more information, see the DB2 documentation.

Configuring dynamic workload broker for FIPS

About this task

If you are using the dynamic workload broker component in your network, perform the following configurations:

- Configure the ita.ini file of every agent that will communicate with the dynamic workload broker component. Ensure that the ssl_port is set and set fips_enable = 1.
- If you are using Dynamic Workload Console, set a secure connection by performing the following:
  1. In Dynamic Workload Console, access dynamic workload broker and expand the **Configuration** menu.
  2. Click **Server Connections**.
  3. In the Server Connections screen, select **Use Secure Connection**.
  4. Click **OK**.

Configuring batch reports for FIPS

About this task

To configure batch reports for FIPS compliance, perform the following steps:

- Import the FIPS certificate from the database server to a Java trustStore on the client. Use the Java keytool utility to import the certificate into the trustStore.

- Edit the SDK java.security file located in the INSTALL_DIR/java/jre/lib/security directory to insert the **IBMJCEFIPS** provider (**com.ibm.crypto.fips.provider.IBMJCEFIPS**). **IBMJCEFIPS** must precede the **IBMJCE** provider in the provider list.

  The following is an example of the edited SDK java.security file:

  ```
  security.provider.1=com.ibm.crypto.fips.provider.IBMJCEFIPS
  security.provider.2=com.ibm.crypto.provider.IBMJCE
  security.provider.3=com.ibm.jsse.IBMJSSEProvider
  security.provider.4=com.ibm.jsse2.IBMJSSEProvider2
  security.provider.5=com.ibm.security.jgss.IBMJGSSProvider
  security.provider.6=com.ibm.security.cert.IBMCertPath
  security.provider.7=com.ibm.crypto.pkcs11.provider.IBMPKCS11
  security.provider.8=com.ibm.security.cmskeystore.CMSProvider
  security.provider.9=com.ibm.security.jgss.mech.spnego.IBMSPNEGO
  ```

  The following is an example of the edited java.security file if you are using the Oracle Java SE Development Kit:

  ```
  security.provider.1=sun.security.provider.Sun
  security.provider.2=com.ibm.crypto.fips.provider.IBMJCEFIPS
  security.provider.3=com.ibm.crypto.provider.IBMJCE
  security.provider.4=com.ibm.jsse.IBMJSSEProvider
  security.provider.5=com.ibm.jsse2.IBMJSSEProvider2
  security.provider.6=com.ibm.security.jgss.IBMJGSSProvider
  security.provider.7=com.ibm.security.cert.IBMCertPath
  security.provider.8=com.ibm.i5os.jsse.JSSEProvider
  #security.provider.8=com.ibm.crypto.pkcs11.provider.IBMPKCS11
  security.provider.9=com.ibm.security.jgss.mech.spnego.IBMSPNEGO
  ```

- Verify that the keystore.type parameter is the same as the value specified for type of the keystore in the config.file. The default value is JKS.

Configuring LDAP for FIPS

About this task

To be FIPS-compliant if you are using an LDAP server, before configuring LDAP, edit the security.xml file. Edit the following value:

```
"com.ibm.ssl.contextProvider" value="IBMJSSEFIPS"
```

Finding the GSKit version on agents running on UNIX and Linux operating systems

About this task

To find which version of GSKit runs on your agent, go to following path depending on the version of GSKit and submit the appropriate command:

GSKit 32 bit

Path /usr/Tivoli/TWS/GSKit32/8/bin

Command
gsk8ver

GSKit 64 bit

Path /usr/Tivoli/TWS/GSKit64/8/bin

Command
gsk8ver_64

On UNIX and Linux, you can optionally run the ita_props.sh script to set the environment to /usr/Tivoli/TWS/GSKit32/8/bin or /usr/Tivoli/TWS/GSKit64/8/

`bin`, so that you can run this command directly without having to specify the relative path.

Chapter 8. Data maintenance

This chapter describes how to maintain your IBM Workload Scheduler database and other data files. The database is hosted on either the DB2 or Oracle RDBMS infrastructure, as you determined when you installed it. You should use the documentation of DB2 or Oracle for general instructions on database maintenance. This chapter describes the maintenance activities that are specific to IBM Workload Scheduler.

It comprises the following sections:

Maintaining the database

This section discusses the following:

- Backing up and restoring files in the IBM Workload Scheduler databases. See "Backing up and restoring."
- Ensuring that a backup master domain manager is as up-to-date as possible. See "Using a backup master domain manager with a backup database."
- Maintaining the performance level of the IBM Workload Scheduler databases. See "Reorganizing the database" on page 357.

Backing up and restoring

To minimize downtime during disaster recovery, back up your master data files frequently to either offline storage or a backup master domain manager.

Backing up the database to offline storage

Run a frequent backup of the database to offline storage. Follow the instructions in the DB2 or Oracle documentation, as appropriate.

IBM Workload Scheduler is supplied with a utility that can be used for backup. It is called **tws_inst_pull_info**. Its primary use is as a tool to gather IBM Workload Scheduler information for IBM Software Support in the event of any problems arising. However, it can equally be used as a backup tool. It backs up the database (DB2 only), the configuration files and the log files.

This tool is described in *IBM Workload Scheduler: Troubleshooting Guide* and gives full details of what files are backed up, how to take a backup, and how to restore from one.

Using a backup master domain manager with a backup database

Set up a backup master domain manager that accesses a different database than the master domain manager, and get your database administrator to set up a

mirror of the master domain manager's database onto the backup master domain manager's database. In this way your backup master domain manager not only receives copies of all the processing messages, as is provided for by the setting of the *FullStatus* attribute on the backup master domain manager, but is also able to access the mirrored database. The mirror frequency must be set high enough to match the frequency with which you change the database.

For more information about how to use a backup master domain manager, see "Changing a domain manager or dynamic domain manager" on page 411.

Backing up the configuration files

The configuration files used by IBM Workload Scheduler are found in the following places:

<*TWA_home*>**/TWS**
: For the user options file, `useropts`.

<*TWA_home*>**/TWS/*.***
: For the `localopts`, `Sfinal`, `Security` and `*.msg` files

<*TWA_home*>**/TWS/mozart/*.***
: This directory contains the following files:

  `runmsgno`
  : This is used for the allocation of unique prompt numbers. On the master domain manager, this file should not be edited manually. On other workstations it can be edited only in the circumstances described in the *IBM Workload Scheduler: Troubleshooting Guide*. This file does not need to be backed up.

  `globalopts`
  : This is used to store a copy of three of the global properties stored in the database. If you have used versions of IBM Workload Scheduler prior to version 8.3 you will probably remember that it was an editable file that contained the global options. It is no longer used for this purpose. It must be edited only in the circumstances described in "Changing a master domain manager" on page 415. This file should be backed up if it is edited.

<*TWA_home*>**/WAS/TWSprofile/properties**
: For the application server configuration file, `TWSConfig.properties`

<*TWA_home*>**/WAS/TWSprofile/config**
: This contains the other configuration files for the WebSphere Application Server. Do not back them up manually. A utility to back them up is described in "Application server - configuration files backup and restore" on page 454.

<*TWA_home*>**/TWS/schedForecast**
: For forecast plan files.

<*TWA_home*>**/TWS/schedlog**
: For archived plan files.

<*TWA_home*>**/TWS/schedTrial**
: For trial plan files.

A detailed list of all files is not supplied, as there are too many files. Back up all the files in these directories.

Note: The **tws_inst_pull_info** tool (described in the *IBM Workload Scheduler: Troubleshooting Guide*) is provided for sending information to support, but can also be used to perform a backup of a DB2 database and some of the configuration files.

Backing up log files

Make a regular offline backup of all log files, identifying them from the information given in the section on log and trace files in the *IBM Workload Scheduler: Troubleshooting Guide*.

If you use **tws_inst_pull_info** for backup (see the documentation in the same guide), you do not need to separately backup these files.

Reorganizing the database

The database requires routine maintenance, as follows:

DB2 The DB2 database has been set up to maintain itself, so there is little user maintenance to do. Periodically, DB2 checks the database by running an internal routine. DB2 determines when this routine must be run using a default policy. This policy can be modified, if need be, or can be switched off so that DB2 does not perform internal automatic maintenance. Using the statistical information that DB2 discovers by running this routine, it adjusts its internal processing parameters to maximize its performance.

This routine has also been made available for you to run manually in the case either where you feel that the performance of DB2 has degraded, or because you have just added a large amount of data , and anticipate performance problems. The routine is imbedded in a tool called **dbrunstats**, which can be run to improve performance while DB2 is processing data without causing any interruption.

It is also possible to physically and logically reorganize the database using the **dbreorg** script. This effectively re-creates the *tablespace* using its internal algorithms to determine the best way to physically and logically organize the tables and indexes on disk. This process is time-consuming, and requires that IBM Workload Scheduler is down while it is run, but it does provide you with a freshly reorganized database after major changes.

The use of these tools is described in "Administrative tasks - DB2" on page 364.

These tools are implementations of standard DB2 facilities. If you are an expert user of DB2 you can use the standard facilities of DB2 to achieve the same results. For details go to the Information Center for DB2, version 9.5, at: http://publib.boulder.ibm.com/infocenter/db2luw/v9r5//index.jsp.

Oracle For Oracle databases see the Oracle maintenance documentation.

Oracle 10g by default has an internally scheduled procedure to collect database statistics: if the default schedule is not changed, Oracle 10g will automatically optimize its performance by running this procedure daily. Oracle 9i does not have the same schedule by default, but could be set up to do so.

Maintaining the file system

Some of the file systems and directories need periodic maintenance. The details are given under the following topics:

- "Avoiding full file systems"
- "Log files and archived files" on page 361
- "Temporary files" on page 363
- "Managing event message queue file sizes" on page 364

Avoiding full file systems

Perhaps the most important maintenance task to perform is that of regularly controlling the file system or systems where IBM Workload Scheduler is installed, particularly on the master domain manager.

IBM Workload Scheduler has a number of files that can grow in size, either with more extensive use, such as the Symphony file, or in the event of network problems, such as the message files. If the Symphony file, in particular, cannot be expanded to contain all the required records, it might become corrupted. If this happens on a fault-tolerant agent or on a domain manager other than the master domain manager, there is a recovery procedure (see the *IBM Workload Scheduler: Troubleshooting Guide*). If the Symphony file on the master domain manager is corrupted, you have no alternative but to restart IBM Workload Scheduler, losing the current plan's workload.

It is thus *most important* that you monitor the available space on the file system of the master domain manager where the Symphony file is generated, to ensure that there is always sufficient space for it to expand to cover any workload peaks, and also that there is sufficient space for message files to expand in the event of network problems. Your experience with your workload and your network will guide you to determine what are the acceptable limits of available disk space.

The approximate size of the Symphony file can be estimated in advance. It contains items related both to the plan (see Table 71) and to the database (see Table 72 on page 359). Estimate how many items you have in each category, multiply them by the indicated size in bytes, and sum them to find the approximate Symphony file size:

Table 71. Algorithm for calculating the approximate size of the plan data in the Symphony file

Data in Symphony file from the current plan	Bytes per instance
Per Job Scheduler instance:	512
Per job instance:	512
Per job "docommand" string > 40 bytes:	The length of the "docommand" string
Per ad hoc prompt:	512
Per file dependency:	512
Per recovery prompt:	512
Per recovery job:	512

Table 72. Algorithm for calculating the approximate size of the database data in the Symphony file

Data in Symphony file from the database (on the master domain manager)	**Bytes per instance**
Per workstation:	512
Per resource:	512
Per user:	256
Per prompt:	512
If the global option `ignoreCalendars` is set to *off*, per calendar:	512

If you find that disk space is becoming too limited, and you cannot dynamically extend it, you must create a backup master domain manager with much more space on its file system and then use the **switchmgr** command so that the backup becomes your new domain manager. Instructions on how to do this for any domain manager are given in "Changing a domain manager or dynamic domain manager" on page 411, and in particular for a master domain manager, in "Changing a master domain manager" on page 415.

Monitoring the disk space used by IBM Workload Scheduler

You can use event-driven workload automation (EDWA) to monitor the disk space used by IBM Workload Scheduler and to start a predefined set of actions when one or more specific events take place. You can use EDWA to monitor the used disk space, to verify that there is enough space to generate the Symphony and log files, and to allow the product to work correctly. For more information about event-driven workload automation, see IBM Workload Scheduler User's Guide and Reference.

The following .XML file contains the definition of a sample event rule to monitor the disk filling percentage. This event rule calls the MessageLogger action provider to write a message in a log file in an internal auditing database. If the condition described in the rule is already existing when you deploy the rule, the related event is not generated. For more information about the MessageLogger action provider, see IBM Workload Scheduler User's Guide and Reference :

```
<?xml version="1.0"?>
<eventRuleSet  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
  xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules
   http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules/EventRules.xsd">
 <eventRule name="FILESYSTEMFULL" ruleType="filter" isDraft="yes">
  <eventCondition name="twsDiskMonEvt1" eventProvider="TWSApplicationMonitor" eventType="TWSDiskMonitor">
   <scope>
    * Disk is filling up
   </scope>
   <filteringPredicate>
    <attributeFilter name="FillingPercentage" operator="ge">
     <value>filling_percentage</value>
    </attributeFilter>
    <attributeFilter name="Workstation" operator="eq">
     <value>workstation_name</value>
    </attributeFilter>
    <attributeFilter name="SampleInterval" operator="eq">
     <value>sample_interval</value>
    </attributeFilter>
    <attributeFilter name="MountPoint" operator="eq">
     <value>mount_point</value>
    </attributeFilter>
```

```
  </filteringPredicate>
 </eventCondition>
 <action actionProvider="MessageLogger" actionType="MSGLOG" responseType="onDetection">
  <scope>
   OBJECT=ADWDAD MESSAGE=Disk is filling up
  </scope>
  <parameter name="ObjectKey">
   <value>object_key</value>
  </parameter>
  <parameter name="Severity">
   <value>message_severity</value>
  </parameter>
  <parameter name="Message">
   <value>log_message</value>
  </parameter>
 </action>
</eventRule>
```

where:

filling_percentage
Is the filling percentage. Supported operators are as follows:

ge causes the event generation when the disk filling percentage increases over the threshold value. The event is generated only the first time the specified disk filling percentage is reached. If you restart the SSM agent and the filling percentage is higher than the threshold value, the event is generated again. Table 73 provides an example in which the **ge** operator is set to 70%.

Table 73. Example for the ge operator

Mailbox name	Filling percentage	Action
Sample (0)	>= 70%	event not generated
Sample (0)	< 70%	event not generated
Sample (n-1)	< 70%	event not generated
Sample (n)	>= 70%	event generated
Sample (n+1)	>= 70%	event not generated

le causes the event generation when the disk filling percentage decreases under the threshold value. The event is generated only the first time the specified disk filling percentage is reached. If you restart the SSM agent and the filling percentage is lower than the threshold value, the event is not generated until the filling percentage increases over the threshold value and then decreases under it again. Table 74 provides an example in which the **le** operator is set to 50%:

Table 74. Example for the le operator

Mailbox name	Filling percentage	Action
Sample (0)	<= 50%	event not generated
Sample (0)	> 50%	event not generated
Sample (n-1)	> 50%	event not generated
Sample (n)	<= 50%	event generated
Sample (n+1)	<= 50%	event not generated

workstation_name
Is the workstation on which the event is generated.

sample_interval
Is the interval, expressed in seconds, for monitoring the disk filling percentage.

mount_point
Is the mount point of the file system where IBM Workload Scheduler is installed, for example: "C:" on Windows systems or "/" on UNIX systems.

object_key
Is a key identifying the object to which the message pertains.

message_severity
Is the severity of the message.

log_message
Is the message to be logged.

Log files and archived files

Log files are produced from a variety of IBM Workload Scheduler activities. Other activities produce files which are archived after they have been used. The details are given in Table 75:

Table 75. Log and trace file maintenance

Activity	Description	Location	Maintenance method
Fault-tolerant agent	Each IBM Workload Scheduler process logs its activities, writing them in log and trace message files:		
	Log messages These are messages intended for use directly by you, and provide information, errors and warnings about the processes.	**netman** <*TWA_home*>/TWS/stdlist/logs/ <*yyyymmdd*>_NETMAN.log **Other processes** <*TWA_home*>/TWS/stdlist/logs/ <*yyyymmdd*>_TWSMERGE.log This is the default situation. You can set an option in the `localopts` file to create separate log files for the major processes.	**rmstdlist**
	Trace messages These are messages written when a problem occurs that you can probably not solve without the assistance of IBM Software Support.	**netman** <*TWA_home*>/TWS/stdlist/traces/ <*yyyymmdd*>_NETMAN.log **Other processes** <*TWA_home*>/TWS/stdlist/traces/ <yyyymmdd>_TWSMERGE.log This is the default situation. You can set an option in the `localopts` file to create separate trace files for the major processes.	
Master domain manager job management	The job manager process on the master domain manager archives the previous period's `Symphony` file.	<*TWA_home*>/TWS/schedlog	Manual

Table 75. Log and trace file maintenance (continued)

Activity	Description	Location	Maintenance method
Job	Each job that runs under IBM Workload Scheduler control creates an output file. These files are archived.	*<TWA_home>*/TWS/stdlist/*<date>* where *<date>* is in the format yyyy.mm.dd	**rmstdlist**
Dynamic agent	**Log messages**	**Windows** <*TWA_home*>\TWS\stdlist\JM\JobManager_message.log **UNIX** <*TWA_home*>/TWS/stdlist/JM/JobManager_message.log	Regular housekeeping is performed through the configuration of several parameters. See "Regular maintenance" on page 71.
	Trace messages	**Windows** • <*TWA_home*>\TWS\stdlist\JM\ITA_trace.log • <*TWA_home*>\TWS\stdlist\JM\JobManager_trace.log • <*TWA_home*>\TWS\JavaExt\logs\javaExecutor0.log **UNIX** • <*TWA_home*>/TWS/stdlist/JM/ITA_trace.log • <*TWA_home*>/TWS/stdlist/JM/JobManager_trace.log • <*TWA_home*>/TWS/JavaExt/logs/javaExecutor0.log	
	Jobs with advanced options	**Windows** <*TWA_home*>\TWS\stdlist\JM\<*date*> **UNIX** <*TWA_home*>/TWS/stdlist/JM/<*date*> where <*date*> is in the format yyyy.mm.dd	
Forecast and trial plan creation	The creation of forecast and trial plans require manual maintenance.	**Forecast plan** These files are to be maintained manually **Trail plan** These files are to be maintained manually	Manual
Audit	The audit facility writes log files.	<*TWA_home*>/TWS/audit	Manual

Table 75. Log and trace file maintenance (continued)

Activity	Description	Location	Maintenance method
DB2 UDB	DB2 logs its activities.	Information about the location and viewing method for DB2 log files is supplied in the DB2 documentation, in the Knowledge Center for DB2. The main file to control is the `db2diag.log` file, which is the most important DB2 diagnostic file, which, without intervention, grows endlessly with no reuse of wasted space. This does not apply, however, to the database log files used by IBM Workload Automation, which are set up for circular reuse of disk space, so they don't grow in size over a maximum value.	See the DB2 documentation.
Oracle database	Oracle logs its activities.	See the Oracle documentation.	See the Oracle documentation.
The WebSphere Application Server	The application server writes log files.	*<WAS_profile_path>*`/logs` where the default value of *WAS_profile_path* is *<TWA_home>*`/WAS/TWSprofile`	Manual

Netcool SSM monitoring agent (not supported on IBM i systems)	The agent writes log files.	*<TWA_home>*`/ssm/Log/`
		`ssmagent.log`
		`traps.log`
			Manual

Other	Other activities also write trace and log files.	*<TWA_home>*`/TWS/methods`	Manual

The easiest method of controlling the growth of these directories is to decide how long the log files are needed, then schedule a IBM Workload Scheduler job to remove any files older than the given number of days. Use the **rmstdlist** command for the process and job log files, and use a manual date check and deletion routine for the others. Make sure that no processes are using these files when you perform these activities.

See the *IBM Workload Scheduler: User's Guide and Reference* for full details of the **rmstdlist** command.

Note: The **rmstdlist** command might give different results on different platforms for the same scenario. This is because on UNIX platforms the command uses the *–mtime* option of the **find** command, which is interpreted differently on different UNIX platforms.

Temporary files

The IBM Workload Scheduler master domain manager uses temporary files, located in *<TWA_home>*`/TWS/tmp` or `/tmp` and named `TWS`*<XXXX>*, when compiling new production control databases. These files are deleted when compiling is complete.

This directory also contains the IBM Workload Scheduler installation files and log files.

Managing event message queue file sizes

This publication contains the following information with respect to managing event message queue file sizes:

- See "Planning space for queues" on page 287 to learn about planning space for message event queues (and also how to use **evtsize** to resize the queues
- See "Managing the event processor" on page 444 to learn about managing the EIF event queue
- See "Disk Space" on page 491 to learn about the impacts that increased fault tolerance can have on message queues
- See "Workload spreading" on page 488 to learn about how to avoid bottlenecks in the `Mailbox.msg` queue.

Administrative tasks - DB2

This section describes how to perform some specific administrative tasks on DB2, as follows:

- "Changing DB2 passwords"
- "Locating the DB2 tools"
- "User permissions for running the DB2 tools" on page 365
- "Administering the DB2 maintenance feature" on page 365
- "Reorganizing the DB2 database" on page 367
- "Monitoring the lock list memory" on page 368

Changing DB2 passwords

About this task

To change passwords used by DB2 other than the *<TWS_user>* password or the passwords of the user IDs used by IBM Workload Scheduler to access the database (see "Changing key IBM Workload Scheduler passwords" on page 419) follow the instructions in the DB2 documentation; they do not directly impact IBM Workload Scheduler.

Locating the DB2 tools

About this task

IBM Workload Scheduler is supplied with a small set of tools that you use to perform the following administrative tasks for DB2:

- Run the DB2 statistics program, to maximize the performance of DB2 (dbrunstats). See "Running DB2 maintenance manually" on page 366 for a full description of how to use the tool.
- Reorganize the database (dbreorg). See "Reorganizing the DB2 database" on page 367 for a full description of how to use the tool.

Find these tools in the following directory:

<TWA_home>`/TWS/dbtools/db2/scripts`

Note: The tools in this directory include some that are for the use of IBM Software Support:

dbcatalog
dbsetup

Do not run these scripts. To do so might damage or overwrite the data in your database.

User permissions for running the DB2 tools

The DB2 tools must be run by a user who has the following permissions:

- DB2 administrator permissions – the user must be defined to DB2 as a DB2 Administrator
- Full access (777) to the IBM Workload Scheduler installation directory

Administering the DB2 maintenance feature

At installation, DB2 automatic maintenance is switched on, which means that DB2 periodically checks to see if it needs to collect new database statistics, so that it can perform the maintenance, adjusting the performance parameters to maximize performance.

This section describes how to administer the automatic maintenance, by changing how and when it is run, switching it off and on again, and running it manually. See the following:

- "Modifying the DB2 automatic maintenance policy"
- "Switching off automatic maintenance"
- "Switching on automatic maintenance" on page 366
- "Running DB2 maintenance manually" on page 366

Modifying the DB2 automatic maintenance policy

To know when and how the statistics used by the automatic maintenance must be collected, DB2 uses a default policy, which can be customized. The procedure is as follows:

1. Right-click the database in the DB2 Control Center and select **Configure Automatic Maintenance** from the menu.
2. Follow the instructions in the wizard, modifying any of the default policy parameters that you think might improve the way DB2 chooses when to run the automatic maintenance.

Switching off automatic maintenance

If you want to take full manual control of the database, switch off the automatic maintenance as follows:

1. Check that the user who is going to run the procedure has the appropriate rights (see "User permissions for running the DB2 tools")
2. On the DB2 server computer, open a DB2 shell, as follows:

   UNIX Follow these steps:

   a. Issue the command **`su - db2inst1`**, or change to the subdirectory `sqllib` of the home directory of the owner of the DB2 instance (by default *db2inst1*)

   b. Launch the command **`. ./db2profile`**

   Windows

   Select from the **Start** menu, **Programs** → **IBM DB2** → **Command Line Tools** → **Command Window**
3. Check that the command shell is correctly initialized by issuing the command **`db2`**, and checking that the command is recognized.
4. Issue the command **`quit`** to leave the DB2 Processor mode.
5. Issue the following command:

db2 UPDATE DB CFG FOR <database_name> USING AUTO_MAINT OFF

where *<database_name>* is the name of the IBM Workload Scheduler database (the installed default name is *TWS*; supply this value unless you have changed it).

6. To make the changes effective, either disconnect and reconnect all the DB2 clients, or restart the DB2 instance (using **db2stop** and **db2start**).

Switching on automatic maintenance

To turn the automatic maintenance back on again, do as follows:

1. Check that the user who is going to run the procedure has the appropriate rights (see "User permissions for running the DB2 tools" on page 365)
2. On the DB2 server computer, open a DB2 shell, as follows:

   UNIX Follow these steps:

   a. Issue the command **su - db2inst1**, or change to the subdirectory `sqllib` of the home directory of the owner of the DB2 instance (by default *db2inst1*)

   b. Launch the command **. ./db2profile**

   Windows

   Select from the **Start** menu, **Programs** → **IBM DB2** → **Command Line Tools** → **Command Window**
3. Check that the command shell is correctly initialized by issuing the command **db2**, and checking that the command is recognized.
4. Issue the command **quit** to leave the DB2 Processor mode.
5. Issue the following command:

   db2 UPDATE DB CFG FOR <database_name> USING AUTO_MAINT ON

   where *<database_name>* is the name of the IBM Workload Scheduler database (the installed default name is *TWS*; supply this value unless you have changed it).
6. To make the changes effective, either disconnect and reconnect all the DB2 clients, or restart the DB2 instance (using **db2stop** and **db2start**).

Running DB2 maintenance manually

This section describes how to perform the DB2 maintenance process on demand, instead of waiting for DB2 to do it according to its automatic maintenance policy. The process is run by the tool **dbrunstats** which you can run whenever you need to, without stopping DB2 or interrupting its processing.

To run this tool, follow this procedure:

1. Locate the DB2 tools: see "Locating the DB2 tools" on page 364.
2. Check that the user who is going to run the procedure has the appropriate rights (see "User permissions for running the DB2 tools" on page 365)
3. Open a DB2 shell, as follows:

   UNIX Follow these steps:

   a. Issue the command **su - db2inst1**, or change to the subdirectory `sqllib` of the home directory of the owner of the DB2 instance (by default *db2inst1*)

   b. Launch the command **. ./db2profile**

   Windows

   Select from the **Start** menu, **Programs** → **IBM DB2** → **Command Line Tools** → **Command Window**

4. Check that the command shell is correctly initialized by issuing the command **db2**, and checking that the command is recognized.
5. Issue the command **quit** to leave the DB2 Processor mode.
6. From within the shell, change to the directory *<TWA_home>*/TWS/dbtools7db27scripts for master domain managers, or to the directory *<TWA_home>*/TDWB/dbtools7db27scripts for dynamic domain managers.
7. Run the script:

   UNIX **dbrunstats.sh database [user [password]]**

   Windows
   dbrunstats database [user [password]]

   where:

   database
   The name of the database:
   - If you are running this from the computer where the DB2 server is installed, the installed default name is *TWS*. Supply this value unless you have changed it.
   - If you are running this from the computer where the DB2 client is installed, the installed default name is *TWS_DB*. Supply this value unless you have changed it.

   user The DB2 administration user. If this is omitted the ID of the user running the command will be used.

   password
   The password of the DB2 administration user. If this is omitted it will be requested interactively.

   The script runs, giving you various messages denoting its progress and successful conclusion. At the end (it is not particularly time-consuming) the database performance parameters have been reset to maximize performance.

Reorganizing the DB2 database

About this task

Using this tool, the database physically reorganizes the data tables and indexes, optimizing disk space usage and ease of data access. The process is time-consuming, requires that the database is backed up, and that IBM Workload Scheduler is stopped. However, at the end you have a database that is completely reorganized.

To reorganize the database follow this procedure:

1. Back up the IBM Workload Scheduler database. Use the method described in "Backing up the database to offline storage" on page 355.
2. Stop WebSphere Application Server and appservman by running the following command:

   ```
   conman "stopappserver;wait"
   ```

   See "Starting and stopping the application server and **appservman**" on page 450 for full details.
3. Check that the user who is going to run the procedure has the appropriate rights (see "User permissions for running the DB2 tools" on page 365)
4. Open a DB2 shell, as follows:

   UNIX Follow these steps:

a. Issue the command **su - db2inst1**, or change to the subdirectory `sqllib` of the home directory of the owner of the DB2 instance (by default *db2inst1*)
b. Launch the command **. ./db2profile**

Windows
Select from the **Start** menu, **Programs → IBM DB2 → Command Line Tools → Command Window**

5. Check that the command shell is correctly initialized by issuing the command **db2**, and checking that the command is recognized.
6. Issue the command **quit** to leave the DB2 Processor mode.
7. From within the shell, change to the directory <*TWA_home*>/TWS/dbtools7db27scripts for master domain managers, or to the directory <*TWA_home*>/TDWB/dbtools7db27scripts for dynamic domain managers.
8. Run the script:

UNIX **dbreorg.sh database [user [password]]**

Windows
dbreorg database [user [password]]

where:

database
The name of the database:
- If you are running this from the computer where the DB2 server is installed, the installed default name is *TWS*. Supply this value unless you have changed it.
- If you are running this from the computer where the DB2 client is installed, the installed default name is *TWS_DB*. Supply this value unless you have changed it.

user The DB2 administration user. If this is omitted the ID of the user running the command will be used.

password
The password of the DB2 administration user. If this is omitted it will be requested interactively.

The script runs, giving you various messages denoting its progress and successful conclusion.

9. Restart WebSphere Application Server and appservman by running the following command:

```
conman "startappserver;wait"
```

See "Starting and stopping the application server and **appservman**" on page 450 for full details.

Monitoring the lock list memory

About this task

If the memory that DB2 allocates for its lock list begins to be fully used, DB2 can be forced into a "*lock escalation*", where it starts to lock whole tables instead of just individual table rows, and increasing the risk of getting into a deadlock.

This happens especially when there are long transactions, such as the creation or extension of a plan (production, trial, or forecast).

To avoid this problem occurring, set the automatic notification in the DB2 Health Center, so that you can be advised of any lock list problems building up.

However, if you think that deadlock situations have been occurring, follow this procedure to verify:

1. With the WebSphere Application Server active, log on as DB2 administrator to the DB2 server, for example,

   su - db2inst1

2. Run the following command to determine where the IBM Workload Scheduler database is located:

   db2 list active databases

   The output might be as follows:

   ```
   Database name                   = TWS
   Applications connected currently = 2
   Database path                   = /home/db2inst1/db2inst1/NODE0000/SQL00002/
   ```

3. Run:

   cd <Database path>/db2event/db2detaildeadlock

4. Connect to the IBM Workload Scheduler database, for example:

   db2 connect to TWS

5. Flush the event monitor that watches over deadlocks (active by default) with the following:

   db2 flush event monitor db2detaildeadlock

6. Disconnect from the database with:

   db2 terminate

7. Obtain the event monitor output with:

   db2evmon -path . > deadlock.out

   The file `deadlock.out` now contains the complete deadlock history since the previous flush operation.

8. To find out if there have been deadlocks and when they occurred, run:

   grep "Deadlock detection time" deadlock.out

   The output might be as follows:

   ```
   Deadlock detection time: 11/07/2008 13:02:10.494600
   Deadlock detection time: 11/07/2008 14:55:52.369623
   ```

9. But the fact that a deadlock occurred does not necessarily mean that the lock list memory is inadequate. For that you need to establish a relationship with lock escalation. To find out if there have been lock escalation incidents prior to deadlocks, run:

   grep "Requesting lock as part of escalation: TRUE" deadlock.out

   The output might be as follows:

   ```
   Requesting lock as part of escalation: TRUE
   Requesting lock as part of escalation: TRUE
   ```

   If there has been lock escalation related to deadlocks, it is a good idea to modify the values of the following parameters.

   LOCKLIST
   : This configures, in 4KB pages, the amount of memory allocated to locking management

MAXLOCKS

This configures the percentage of the memory that a single transaction can use, above which DB2 escalates, even though the memory might not be full

10. To determine the values currently being applied to the IBM Workload Scheduler database, do the following:

    db2 get db cfg for TWS | grep LOCK

    The output might be as follows:

    ```
    Max storage for lock list (4KB)             (LOCKLIST) = 8192
    Percent. of lock lists per application      (MAXLOCKS) = 60
    Lock timeout (sec)                       (LOCKTIMEOUT) = 180
    ```

    The example shows the typical output for the IBM Workload Scheduler database if no modification has taken place to these values:

    - "8192" = 4KB x 8192 pages = 32 MB of memory
    - "60" = 60% – the percentage of memory that a single transaction can occupy before triggering an escalation
    - "180" = 3 minutes of timeout for the period a transaction can wait to obtain a lock

11. The most straightforward action to take is to double the amount of memory to 64MB, which you do with the command:

    db2 update db cfg for TWS using LOCKLIST 16384 immediate

12. Alternatively, you can set DB2 to automatically modify the LOCKLIST and MAXLOCKS parameters according to the amount of escalation being experienced and the available system memory. This self-tuning is a slow process, but adapts the database to the needs of the data and the available system configuration. It is done by setting the values of these parameters to AUTOMATIC, as follows:

    db2 update db cfg for TWS using LOCKLIST AUTOMATIC immediate

    DB2 responds with messages telling you that MAXLOCKS has also been set to AUTOMATIC:

    ```
    SQL5146W "MAXLOCKS" must be set to "AUTOMATIC" when "LOCKLIST" is
    "AUTOMATIC".
    "MAXLOCKS" has been set to "AUTOMATIC"
    ```

    Note: The self-tuning facility is only available from V9.1 of DB2.

Administrative tasks - Oracle

This section describes how to perform some specific administrative tasks for the Oracle database.

Changing the Oracle access password

About this task

This is described as part of the process of changing the password for a master domain manager or backup master domain manager. See "Changing key IBM Workload Scheduler passwords" on page 419.

Locating the Oracle tools

About this task

IBM Workload Scheduler is supplied with a small set of tools that you use to perform the following administrative tasks for Oracle:

- Grant the user permissions for the Dynamic Workload Console views (`dbgrant`). See the Dynamic Workload Console online help for full details.
- Migrating from DB2 to Oracle or vice versa (`prepareSQLScripts`, `createdb_root_ora`, `updateSetupCmdLine`). See "Migrating data from DB2 to Oracle and *vice versa*" on page 372 for full details.

Locate these tools in the following directory:

`<TWA_home>/TWS/dbtools/oracle/scripts`

Note: The directory also includes some scripts that are only for the use of IBM Software Support:

dbmigrate
dbpartition
dbsetup
dbupgrade
launchdb_root_ora
_migratedb_root_ora

Do not run these scripts. To do so might damage or overwrite the data in your database.

Maintaining the Oracle database

Like DB2, Oracle has a routine that regularly maintains the database. Similarly, this too can be run manually. The tool is invoked as follows:

```
dbms_stats.gather_schema_stats<schema_owner>
```

See the Oracle documentation for full details of how and when to run it.

Obtaining information about the IBM Workload Scheduler databases installed on an Oracle instance

About this task

To determine which IBM Workload Scheduler databases are installed on an Oracle instance, do the following:

```
su - oracle (UNIX only)
sqlplus system/<system_password>@<service_name>
SQL> select * from all_tws_schemas;
```

The output should look like the following:

```
SCHEMA_NAME
------------------------------
MDL
mdm85TWS_user
```

Note:

1. More than one instance of IBM Workload Scheduler can be shared in one instance of Oracle, using different schemas.
2. In Oracle, the concept of "schema" and "user" are the same, so dropping an Oracle schema means dropping an Oracle user, which you do as follows:

   ```
   SQL> drop user MDL cascade;
   ```

User permissions for running the Oracle tools

The Oracle tools must be run by a user who has the following permissions:

- Oracle administrator permissions – the user must be defined to Oracle as an administrator
- Full access (777) to the IBM Workload Scheduler installation directory

Migrating data from DB2 to Oracle and *vice versa*

This section applies to IBM Workload Scheduler master domain managers and its backup. It documents how to migrate the IBM Workload Scheduler data from one RDBMS to another.

There are two ways to accomplish the migration. You can use either way to migrate your data from DB2 to Oracle or vice versa.

Parallel data migration
: The migration is between two instances of IBM Workload Scheduler, one that uses DB2 while the other uses Oracle.

Reconfiguration
: The database is migrated from one RDBMS support mechanism to another, and IBM Workload Scheduler instance is re-configured to point to a different database without installing another instance.

Note: Neither of these procedures migrate the following information from the source database:

- The preproduction plan
- The history of job runs and job statistics
- The state of running event rule instances. This means that any complex event rules, where part of the rule has been satisfied prior to the database migration, are generated after the migration as new rules. Even if the subsequent conditions of the event rule are satisfied, the record that the first part of the rule was satisfied is no longer available, so the rule will never be completely satisfied.

Parallel data migration from DB2 to Oracle

About this task

With the following steps all scheduling object definitions and global options can be migrated from the DB2 database of a IBM Workload Scheduler version 9.4 instance to the Oracle database of another instance.

1. Fresh-install another instance of a IBM Workload Scheduler version 9.4 master domain manager and make it point to an Oracle database, by defining the `MDM_ORACLE` as master domain manager workstation name. The installation process automatically defines in the Oracle database the following workstations:
   - `MDM_ORACLE_DWB` is the broker workstation name.
   - `MDM_ORACLE_1` is the agent workstation name.
2. Use **composer** or the Dynamic Workload Console to define this instance as a fault-tolerant agent in the database of the current master domain manager that points to DB2. The fault-tolerant agent workstation name must be `MDM_ORACLE` that you specified in the installation process.
3. On the master domain manager that points to DB2 run the `dataexport` command or script to export all scheduling object definitions and global options from DB2. Find this file in the `bin` subdirectory of the IBM Workload Scheduler home directory.

   Run `dataexport` from a Windows or UNIX command prompt as follows:

   ```
   dataexport <source_dir> <export_dir>
   ```

   where:

   source_dir
   The installation directory of the instance of IBM Workload Scheduler version 9.4 that points to the DB2 database.

   export_dir
   The directory where the export files are to be created.

   For example:

   ```
   dataexport.cmd F:\TWS93\twsDB2user F:\TWS93\export
   ```

   The object definitions and the global options are retrieved from the DB2 database and placed in the `F:\TWS93\export` directory.
4. Verify the following files were created in `export_dir`:
   - calendars.def
   - jobs.def

     Note: The record length supported by DB2 is 4095 characters, but it decreases to 4000 characters with Oracle. When you migrate your job definitions to Oracle, any job scripts or commands exceeding 4000 characters in length are not migrated. In such case, the data import utility replaces the job definition with a dummy job definition and sets the job priority to 0, guaranteeing that successors are not run.
   - `globalOpts.def`
   - `erules.def`
   - `parms.def`
   - `prompts.def`
   - `resources.def`
   - `scheds.def`
   - `topology.def`
   - `users.def` (includes encrypted user passwords)
   - `vartables.def`
   - `rcgroups.def`
   - `acls.def`
   - `sdoms.def`
   - `srols.def`

5. On the master domain manager that points to DB2 do the following:
   a. Ensure that the carry forward option is set to `ALL`. Run:
      ```
      optman chg cf=ALL
      ```
   b. Add the new instance (that you installed in step 1 on page 373 and that you momentarily defined as a fault-tolerant agent) in the current plan. To do this, run:
      ```
      JnextPlan -for 0000
      ```
   c. Check that the new instance was linked by running:
      ```
      conman sc
      ```
6. Open the `export_dir\topology.def` file and substitutes the following values:
   - `MDM_DB2` master domain manager workstation name in the DB2 database with the value `MDM_ORACLE` master domain manager workstation name in the Oracle database.
   - `MDM_DB2_DWB` broker workstation name in the DB2 database with the value `MDM_ORACLE_DWB` broker workstation name in the Oracle database.
   - `MDM_DB2_1` agent workstation name in the DB2 database with the value `MDM_ORACLE_1` agent workstation name in the Oracle database.
7. On the new instance run the `dataimport` command or script to import all scheduling object definitions and global options to the Oracle database. Find this file in the `bin` subdirectory of the IBM Workload Scheduler home directory.

   Run `dataimport` from a Windows or UNIX command prompt as follows:
   ```
   dataimport <source_dir> <export_dir>
   ```

   where:

   source_dir
   The installation directory of the new instance of IBM Workload Scheduler version 9.4 pointing to the Oracle database.

   export_dir
   The directory from where the export files are to be read from. This directory is the same `export_dir` directory specified for `dataexport`.

   For example:
   ```
   dataimport.cmd F:\TWS93\twsORACLEuser F:\TWS93\export
   ```

   The object definitions and the global options are retrieved from the `F:\TWS93\export` directory and stored in the Oracle database.
8. On the master domain manager that points to DB2 run the `conman switchmgr` command to make the IBM Workload Scheduler instance pointing to Oracle as the acting master domain manager. For information on this command see *IBM Workload Scheduler: User's Guide and Reference*.

You have now completed the data migration steps.

Parallel data migration from Oracle to DB2

About this task

With the following steps all scheduling object definitions and global options can be migrated from the Oracle database of a IBM Workload Scheduler version 9.4 instance to the DB2 database of another instance (freshly installed or upgraded to version 9.4).

1. Fresh-install another instance of IBM Workload Scheduler version 9.4 or upgrade an existing instance to version 9.4 making it point to a DB2 database,

by defining the MDM_DB2 as master domain manager workstation name. The installation process automatically defines in the DB2 database the following workstations:

- MDM_DB2_DWB is the broker workstation name.
- MDM_DB2_1 is the agent workstation name.

2. Use composer or the Dynamic Workload Console to define this instance as a fault-tolerant agent in the database of the master domain manager that points to Oracle. The fault-tolerant agent agent workstation name must be MDM_DB2 that you specified in the installation process.
3. On the current master domain manager pointing to the Oracle database run the dataexport command or script to export all scheduling object definitions and global options. Find this file in the bin subdirectory of the IBM Workload Scheduler home directory.

   Run dataexport from a Windows or UNIX command prompt as follows:

   ```
   dataexport <source_dir> <export_dir>
   ```

   where:

   source_dir
   The installation directory of the instance of IBM Workload Scheduler that points to the Oracle database.

   export_dir
   The directory where the export files are to be created.

   For example:

   ```
   dataexport.cmd F:\TWS93\twsORACLEuser F:\TWS93\export
   ```

   The object definitions and the global options are retrieved from the Oracle database and placed in the F:\TWS93\export directory.
4. Verify the following files were created in export_dir:
   - calendars.def
   - erules.def
   - jobs.def
   - globalOpts.def
   - parms.def
   - prompts.def
   - resources.def
   - scheds.def
   - topology.def
   - users.def (includes encrypted user passwords)
   - vartables.def
   - rcgroups.def
   - acls.def
   - sdoms.def
   - srols.def
5. On the current master domain manager that points to Oracle do the following:
   a. Ensure that the carry forward option is set to ALL. Run:

      ```
      optman chg cf=ALL
      ```

   b. Add the new instance (that you installed in step 1 on page 374 and that you momentarily defined as a fault-tolerant agent) in the current plan. To do this, run:

      ```
      JnextPlan -for 0000
      ```

   c. Check that the new instance was linked by running:

```
conman sc
```

6. Open the `export_dir\topology.def` file and substitutes the following values:
   - `MDM_ORACLE` master domain manager workstation name in the Oracle database with the value `MDM_DB2` master domain manager workstation name in the DB2 database.
   - `MDM_ORACLE_DWB` broker workstation name in the Oracle database with the value `MDM_DB2_DWB` broker workstation name in the DB2 database.
   - `MDM_ORACLE_1` agent workstation name in the Oracle database with the value `MDM_DB2_1` agent workstation name in the DB2 database.
7. On the new instance run the `dataimport` command or script to import all scheduling object definitions and global options to DB2. Find this file in the `bin` subdirectory of the IBM Workload Scheduler home directory.

   Run `dataimport` from a Windows or UNIX command prompt as follows:

   ```
   dataimport <source_dir> <export_dir>
   ```

   where:

   source_dir
   The installation directory of the instance of IBM Workload Scheduler that points to the DB2 database.

   export_dir
   The directory from where the export files are to be read from. This directory is the same `export_dir` directory specified for `dataexport`.

   For example:

   ```
   dataimport.cmd F:\TWS93\twsDB2user F:\TWS93\export
   ```

   The object definitions and the global options are retrieved from the `F:\TWS93\export` directory and stored in the DB2 database.
8. On the master domain manager that points to Oracle run the `conman switchmgr` command to make the IBM Workload Scheduler instance pointing to DB2 as the acting master domain manager. For information on the `switchmgr` command see *User's Guide and Reference*.

You have now completed the data migration steps.

Reconfiguration from DB2 to Oracle

About this task

Perform the following steps to migrate all scheduling object definitions and global options from the DB2 database of a IBM Workload Scheduler version 9.4 master domain manager and make them point to an Oracle database.

1. To export all scheduling object definitions and global options from DB2, from a Windows or UNIX command prompt run the `dataexport` command or script located in the `<TWA_home>\bin` directory:

   ```
   dataexport <source_dir> <export_dir>
   ```

   where:

   source_dir
   The IBM Workload Scheduler version 9.4 installation directory.

   export_dir
   The directory where the export files are to be created.

For example:

```
dataexport.cmd F:\TWS93\tws93user F:\TWS93\tws93user\export
```

The object definitions and the global options are retrieved from the DB2 database and stored into the `F:\TWS93\tws93user\export` directory.

2. Verify that the following files were created in `export_dir`:
   - calendars.def
   - erules.def
   - jobs.def
   - globalOpts.def
   - parms.def
   - prompts.def
   - resources.def
   - scheds.def
   - topology.def
   - users.def (includes encrypted user passwords)
   - vartables.def
   - rcgroups.def
   - acls.def
   - sdoms.def
   - srols.def
3. Stop the WebSphere Application Server by using the **conman stopappserver** command (for details, see "Starting and stopping the application server and **appservman**" on page 450).
4. To create an Oracle schema named `tws93user` on the `TWS` database see the section about creating IBM Workload Scheduler SQL tables for Oracle in *Planning and Installation Guide*..

   The names of the table spaces are the defaults: `USERS` and `TEMP`.If an error occurs and you have to rerun this step after fixing it, you must first log in to the database as the administrator, then drop the *twsDbUser* (`tws93user` in the example).
5. Change the data source properties from DB2 to Oracle by using the `changeDataSource.bat (.sh)` command or script to switch the data source in WebSphere Application Server from DB2 to Oracle.

   See "Changing data source properties" on page 432 for details on how to use the command.

   a. Clear the values of the following properties:

      ```
      DB2Type4JndiName
      DB2Type4DatabaseName
      DB2Type4ServerName
      DB2Type4PortNumber
      ```

      Note: For the property `DB2Type4JndiName` replace the value with any character or string to nullify it.

   b. Set these properties to the following values:

      ```
      OracleType2JndiName=jdbc/twsdb
      OracleType2DatabaseName=the Oracle instance name
      OracleType2PortNumber=the Oracle listener port number
      ```

   c. Set the JDBC driver path for the Oracle database to `ORACLE_JDBC_DRIVER_PATH=Oracle_home/jdbc/lib` and the Oracle instance type and name to `ORACLETYPE2URL=JDBC:ORACLE:OCI:@instance_name`
6. Use the `changeSecurityProperties.bat (.sh)` command or script to change the following security settings (see "Changing the security settings" on page 443 for details on using the command):

j2cUserid
Write the value you used for `twsDbUser` in `prepareSQLScripts`.

j2cPassword
Write the password for the `twsDbUser`.

Note: After you updated these two values, make sure that you also erase all the other lines in the security properties file before you export it again with the `changeSecurityProperties` command. Failure to do so will result in all the passwords contained in the file being saved as strings of asterisks (*).

7. Modify the `TWSConfig.properties` file located in the *<WAS_profile_path>*/`properties` directory, where *WAS_profile_path* corresponds to the WebSphere Application Server profile path you specified at installation time. The default path is: *<TWA_home>*/WAS/TWSprofile. Take the comment marks off the following lines and edit them as shown:

```
com.ibm.tws.dao.rdbms.rdbmsName = Oracle
com.ibm.tws.dao.rdbms.modelSchema = <twsDbUser>
com.ibm.tws.dao.rdbms.eventRuleSchema=<twsDbUser>
com.ibm.tws.dao.rdbms.logSchema=<twsDbUser>
```

where `twsDbUser` is the owner of the IBM Workload Scheduler schema that you specified also in the previous steps.

8. For UNIX only: to set the paths for the new database in the WebSphere Application Server profile, run the `updateSetupCmdLine.sh` command or script located in the `<TWA_home>/TWS/dbtools/oracle/scripts` directory as follows:

```
updateSetupCmdLine.sh -installRoot <TWA_home> -dbRoot <DB_home>
```

where:

–installRoot *<TWA_home>*
The installation directory of IBM Workload Scheduler.

–dbRoot *<DB_home>*
The installation directory of the database.

9. Start the WebSphere Application Server by using the **conman startappserver** command (for details, see "Starting and stopping the application server and **appservman**" on page 450)
10. Manually copy the master domain manager definition from the `topology.def` file you exported in step 1 on page 376 (you can find it in `export_dir`) and use `composer new` to add it in the Oracle database.
11. To import all scheduling object definitions and global options to the Oracle database, from a UNIX or Windows command prompt run the `dataimport` file located in `<TWA_home>\bin` as follows:

```
dataimport <source_dir> <export_dir>
```

where:

source_dir
The IBM Workload Scheduler installation directory.

export_dir
The directory from where the export files are to be read from. This directory is the same `export_dir` directory specified for `dataexport`.

For example:

```
dataimport.cmd F:\TWS93\tws93user F:\TWS93\tws93user\export
```

The object definitions and the global options are retrieved from the `F:\TWS93\tws93user\export` directory and stored in the new Oracle database.

12. Run the following command to set the carry forward option to ALL:

    ```
    optman chg cf=ALL
    ```

13. Update the Symphony file by creating a plan with 0 extension period that begins at the end of the current plan:

    ```
    JnextPlan -from start_time -for 0000
    ```

    where *start_time* is the date and time when the current plan ends.

To complete the reconfiguration process, you must also perform the steps listed in the IBM Workload Scheduler: Planning and Installation, Part "IBM Workload Scheduler", Chapter "Creating or upgrading the IBM Workload Scheduler database tables before installing or upgrading", section "Creating or upgrading the database tables if you are using Oracle", subsection "Running scripts to create or upgrade the SQL tables for Oracle", subsection "Procedure to create the IBM dynamic workload broker SQL tables for Oracle".

You have now completed the reconfiguration steps.

To migrate a backup master domain manager, perform the following steps:

- Stop the WebSphere Application Server by using the **conman stopappserver** command (for details, see "Starting and stopping the application server and **appservman**" on page 450)
- Run steps 5 on page 377, 6 on page 377, 8 on page 378
- Start the WebSphere Application Server by using the **conman startappserver** command (for details, see "Starting and stopping the application server and **appservman**" on page 450)
- Set the `isBackupManager` parameter of the `createdb_root` command (script) to `TRUE`.

Reconfiguration from Oracle to DB2

About this task

With the following steps all scheduling object definitions and global options can be migrated from the Oracle database of a IBM Workload Scheduler version 9.4 master domain manager and made to point to a DB2 database.

1. Run the `dataexport` command or script to export all scheduling object definitions and global options from Oracle. Find this file in the `bin` subdirectory of the IBM Workload Scheduler version 9.4 home directory.

   Run `dataexport` from a Windows or UNIX command prompt as follows:

   ```
   dataexport <source_dir> <export_dir>
   ```

   where:

   source_dir
   The IBM Workload Scheduler version 9.4 installation directory.

   export_dir
   The directory where the export files are to be created.

   For example:

   ```
   dataexport.cmd F:\TWS93\tws93user F:\TWS93\tws93user\export
   ```

The object definitions and the global options are retrieved from the Oracle database and placed in the `F:\TWS93\tws93user\export` directory.

2. Verify the following files were created in `export_dir`:
   - calendars.def
   - erules.def
   - jobs.def
   - globalOpts.def
   - parms.def
   - prompts.def
   - resources.def
   - scheds.def
   - topology.def
   - users.def (includes encrypted user passwords)
   - vartables.def
   - rcgroups.def
   - acls.def
   - sdoms.def
   - srols.def
3. Stop WebSphere Application Server as described in "Application server - starting and stopping" on page 447.
4. Run `prepareSQLScripts.bat (.sh)` to customize the SQL scripts with the parameters needed to create the IBM Workload Scheduler database in DB2. Find this file in the *TWA_home*`/TWS/dbtools/db2/scripts` directory.

   Run `prepareSQLScripts` from a Windows or UNIX command prompt as follows:

   - From a UNIX shell run:

     ```
     prepareSQLScripts
        -dbRoot <dbRoot>
        -dbName <dbName>
        -dbLocalAdmin <dbLocalAdmin>
        -twsDbUser <twsDbUser>
        [-tempDir <tempDir>]
        [-dataTablespace <dataTablespace_name>]
        [-dataTablespacePath <dataTablespacePath>]
        [-logTablespace <logTablespace_name>]
        [-logTablespacePath <logTablespacePath>]
        [–tempTablespace <tempTablespace_name>]
        [–userTempTablespace <userTempTablespace_name>]
        [-companyName <companyName>]
        [-masterDmName <masterDmName>]
        [-eifPort <eifPort>]
     ```

   - From a Windows command prompt run:

     ```
     cmd /K prepareSQLScripts
        -dbRoot <dbRoot>
        -dbName <dbName>
        -dbLocalAdmin <dbLocalAdmin>
        -twsDbUser <twsDbUser>
        [-tempDir <tempDir>]
        [-dataTablespace <dataTablespace_name>]
        [-dataTablespacePath <dataTablespacePath>]
        [-logTablespace <logTablespace_name>]
        [-logTablespacePath <logTablespacePath>]
        [–tempTablespace <tempTablespace_name>]
        [–userTempTablespace <userTempTablespace_name>]
        [-companyName <companyName>]
        [-masterDmName <masterDmName>]
        [-eifPort <eifPort>]
     ```

   where:

dbRoot
The path where the RDBMS software is installed.

dbName
The name of the database.

dbLocalAdmin
The user ID of the local database administrator.

twsDbUser
The database user for IBM Workload Scheduler.

tempDir
The directory where the temporary files created by this process are placed. On Windows the default is *<drive>*\Documents and Settings*<current_user>*\Local Settings\Temp.

dataTablespace
The name of the table space for the IBM Workload Scheduler data. The default is TWSDATA. If you provided a different value at installation time, enter that value again.

dataTablespacePath
The path of the table space for the IBM Workload Scheduler data.

logTablespace
The name of the table space for the IBM Workload Scheduler log. The default is TWSLOG. If you provided a different value at installation time, enter that value again.

logTablespacePath
The path of the table space for the IBM Workload Scheduler log files.

tempTablespace
The name of the table space for temporary data. The default is TEMP. If you provided a different value at installation time, enter that value again.

userTempTablespace
The name of the table space for temporary user data. The default is USERTEMP. If you provided a different value at installation time, enter that value again.

companyName
The name of your company. The default is MYCOMPANY. If you provided a different value at installation time, enter that value again.

masterDmName
The name of the master domain. The default is MASTERDM. If you provided a different value at installation time, enter that value again.

eifPort
The EIF port. The default is 31123. If you provided a different value at installation time, enter that value again.

For example:

```
cmd /K prepareSQLScripts.bat -dbRoot D:\DB2
                      -dbName TWS
                      -dbLocalAdmin db2admin
                      -twsDbUser tws93User
```

The SQL scripts are customized to create a database named TWS in DB2 for user tws93user. The names of the table spaces are the defaults: TWSDATA, TWSLOG, TEMP and USERTEMP.

5. Run `createdb_root.bat (.sh)` to create the database following the specifications of the previous step. Find this file in the *tempDir*`/TWA/tws93/scripts` directory, where *tempDir* is the parameter you specified in step 4 on page 380.

   Run `createdb_root` as follows:

   - From a UNIX shell, run:

     ```
     createdb_root
        <dbName>
        <isClientInstallation>
        <dbNodeName>
        <hostName>
        <srvPortNumber>
        <db2Admin>
        <db2AdminPwd>
        <instanceName>
        <isBackupManager>
     ```

   - From a Windows command prompt, run:

     ```
     cmd /K createdb_root
        <dbName>
        <isClientInstallation>
        <dbNodeName>
        <hostName>
        <srvPortNumber>
        <db2Admin>
        <db2AdminPwd>
        <instanceName>
        <isBackupManager>
     ```

   where:

   dbName
   The name of the DB2 database. The maximum length is 5 characters.

   isClientInstallation
   The value is:
   - `TRUE` if the database is a DB2 client.
   - `FALSE` if the database is a DB2 server.

   dbNodeName
   The name of the DB2 node.

   hostName
   The host name of the computer where DB2 is to be installed.

   srvPortNumber
   The TCP/IP port number used to communicate with the DB2 server. The default is 50000.

   db2Admin
   The user ID of the DB2 administrator.

   db2AdminPwd
   The password for `db2Admin`.

   instanceName
   The name of the DB2 server instance.

   isBackupManager
   Specify `TRUE` if you are migrating a backup master domain manager. Specify `FALSE` otherwise.

   For example:

   ```
   createdb_root TWS FALSE TWS_ND myhost 50000 db2admin passw1rd DB2 FALSE
   ```

creates a database named TWS on a DB2 server instance named DB2.

6. Use the changeDataSource.bat (.sh) command or script to switch the data source in WebSphere Application Server from Oracle to DB2.

   See "Changing data source properties" on page 432 for details on how to use the command.

   a. Clear the following properties:

      ```
      OracleType2JndiName
      OracleType2DatabaseName
      OracleType2ServerName
      OracleType2PortNumber
      ```

   b. Set the following properties:

      ```
      DB2Type4JndiName
      DB2Type4DatabaseName
      DB2Type4ServerName
      DB2Type4PortNumber
      ```

   c. Set the JDBC driver path for the DB2 in both DB2_JDBC_DRIVER_PATH and DB2UNIVERSAL_JDBC_DRIVER_PATH (the path is the same for both properties).

7. Reset to a name of your choice the ...JndiName property of the RDBMS from which you are changing.
8. Set to jdbc/twsdb the ...JndiName property of the new RDBMS
   - See that the following properties are set:
     – For DB2:

       ```
       DB2Type4JndiName
       DB2Type4DatabaseName
       DB2Type4ServerName
       DB2Type4PortNumber
       ```

9. Run the changeSecurityProperties.bat (.sh) command or script to change the following security settings:

   j2cUserid
   Write the value you used for twsDbUser in prepareSQLScripts.

   j2cPassword
   Write the password for twsDbUser.

   Note: After you updated these two values, make sure that you also erase all the other lines in the security properties file before you export it again with the changeSecurityProperties command. Failure to do so will result in all the passwords contained in the file being saved as strings of asterisks (*).

   See "Changing the security settings" on page 443 for details.

10. Modify the TWSConfig.properties file located in the <*WAS_profile_path*>/properties directory, where, *WAS_profile_path* corresponds to the WebSphere Application Server profile path you specified at installation time. The default path is: *TWA_home*/WAS/TWSprofile. Comment the following four lines:

    ```
    com.ibm.tws.dao.rdbms.rdbmsName = Oracle
    com.ibm.tws.dao.rdbms.modelSchema = <twsDbUser>
    com.ibm.tws.dao.rdbms.eventRuleSchema
    com.ibm.tws.dao.rdbms.logSchema
    ```

    where twsDbUser is the owner of the IBM Workload Scheduler Oracle schema.

11. For UNIX only: run the updateSetupCmdLine.sh command or script to set the paths for the new database. Locate this script in the <TWA_home>/TWS/dbtools/db2/scripts directory. The syntax is as follows:

    ```
    updateSetupCmdLine.sh -installRoot <TWA_home> -dbRoot <DB_home>
    ```

where:

–installRoot <*TWA_home*>
The installation directory of IBM Workload Scheduler.

–dbRoot <*DB_home*>
The installation directory of the database.

12. Start the WebSphere Application Server using the **conman startappserver** command (see "Starting and stopping the application server and **appservman**" on page 450)
13. Manually copy the master domain manager definition from the `topology.def` file you exported in step 1 on page 379 (you can find it in `export_dir`) and use `composer new` to add it in the DB2 database.
14. Run `dataimport` to import all scheduling object definitions and global options to DB2. Find this file in the `bin` subdirectory of the IBM Workload Scheduler home directory.

    Run `dataimport` from a Windows or UNIX command prompt as follows:

    ```
    dataimport source_dir export_dir
    ```

    where:

    source_dir
    The IBM Workload Scheduler installation directory.

    export_dir
    The directory from where the export files are to be read from. This directory is the same `export_dir` directory specified for `dataexport`.

    For example:

    ```
    dataimport.cmd F:\TWS93\tws93user F:\TWS93\tws93user\export
    ```

    The object definitions and the global options are retrieved from the `F:\TWS93\tws93user\export` directory and stored in the new DB2 database.
15. Run the following command to set the carry forward option to ALL:

    ```
    optman chg cf=ALL
    ```
16. Update the Symphony® file by creating a plan with 0 extension period that begins at the end of the current plan:

    ```
    JnextPlan -from start_time -for 0000
    ```

    where *start_time* is the date and time when the current plan ends.

You have now completed the reconfiguration steps.

To migrate a backup master domain manager, perform the following steps :

- Stop the WebSphere Application Server using the **conman stopappserver** command (see "Starting and stopping the application server and **appservman**" on page 450)
- Perform steps 6 on page 383 to 11 on page 383.
- Start the WebSphere Application Server using the **conman startappserver** command (see "Starting and stopping the application server and **appservman**" on page 450)
- Set the `isBackupManager` parameter of the `createdb_root` command (script) to `TRUE`.

Upgrading your database

About this task

If you want to upgrade your database, change the instance owner, or relocate it to a different host, the procedure for upgrading your database, changing the instance owner, or relocating it, is as follows:

1. If you are changing DB2, check the *node directory* and *database directory* and make a note of the current configuration. To do this, issue the following commands at the DB2 command-line:

   ```
   db2 list node directory show detail

   db2 list database directory
   ```

   where the `show detail` attribute is specified to give the full information in the directory.

   Make a note of the displayed details.
2. Stop the application server, using the command

   ```
   stopWas -direct -user <user> -password <password>
   ```
3. Make the upgrade, instance owner change, or relocation, of the database following the instructions from your database supplier.
4. If you have changed the database host, port, or database name, you will need to update the application server's data source properties, as described in "Changing the database host name, port, or database name" on page 431.
5. If you have changed the database access credentials, you will need to update the application server's security properties, as described in "Changing the security settings" on page 443.
6. Reconfigure the database for IBM Workload Scheduler, as follows:

   DB2

   a. Check the *node directory* and *database directory*, as you did in step 1
   b. If necessary, modify the data displayed by these commands to match the data you noted in step 1. If you are not certain of how to do this, contact IBM Software Support for assistance.

   Oracle Check the Oracle Listener and make sure that the service name is correctly specified.
7. Restart the database.
8. Restart the application server, using the command:

   ```
   startWas -direct -user <user> -password <password>
   ```

Auditing facilities

Describes the audit facilities to track changes in the database and the plan, as well as those that track changes to objects involved in dynamic workload scheduling.

In the Dynamic Workload Console, operators and schedulers can review all changes to scheduling objects, both in the database and in the plan, discover which user performed a specific change, and when the change was performed. Admnistrators can request that each user provide a justification when making changes to an object and log this information in audit trails. Application developers and schedulers can compare and restore previous versions of each changed object, and promote the job workflow from development to test or production environments.

For more information, see the section about keeping track of changes in *Dynamic Workload Console User's Guide*.

Audit trails are useful to check enforcement and effectiveness of IT controls, for accountability, and vulnerability and risk analysis. IT organizations can also use auditing of security-related critical activities to aid in investigations of security incidents. When a security incident occurs, audit trails enable analysis of the history of activities (who did what, when, where, and how) that occurred prior to the security incident, so appropriate corrective actions can be taken. For these reasons, audit trails might need to be archived and accessible for years.

Two separate audit trail facilities are provided:

- Database and plan change tracking - see "Database and plan audit"
- Tracking of changes to scheduling objects to support dynamic workload scheduling - see "Dynamic workload scheduling audit" on page 393

Database and plan audit

An auditing option is available to track changes to the database and the plan. It is disabled by default. It is described in these sections:

- "Enabling and storing audit trails"
- "Audit log header format" on page 388
- "Audit log body format" on page 388
- "Sample audit log entries" on page 392

Enabling and storing audit trails

You can maintain audit trails for information stored in the database and in the plan. By default, auditing is enabled. To disable auditing, use the following global options:

enDbAudit
Enables auditing of the information available in the database.

enPlanAudit
Enables auditing of the information available in the plan.

For more information about global options, see "Global options - detailed description" on page 16.

You can store auditing information in a file, in the IBM Workload Scheduler database, or in both. To define in which type of store to log the audit records, use the **auditStore** global option. For more information about global options, see "Global options - detailed description" on page 16. When auditing database information, all the user modifications are logged, including the current definition of each modified database object. If an object is opened and saved, the action is logged even if no modification was made. When auditing plan information, all the user modifications to the plan are logged. Actions are logged whether or not they are successful.

Choose the storage location of audit records according to the type of information you are auditing, whether it is database or plan:

auditing of the information available in the database (enDbAudit global option)
You can track changes to the database in a file, in the database itself, or in both. To define which type of store to log the audit records, use the **auditStore** global option. For more information about global options, see "Global options - detailed description" on page 16. All the user

modifications are logged, including the current definition of each modified database object. If an object is opened and saved, the action is logged even if no modification was made.

auditing of the information available in the plan (enPlanAudit global option)
You can track changes to the plan in a file. When you enable auditing of the information available in the plan, the information is saved to a file. All the user modifications to the plan are logged. Actions are logged whether or not they are successful.

Storing auditing information in a file (auditStore=file)

This storage location is available when you audit information in the database (**enDbAudit** global option) and in the plan (**enPlanAudit** global option). Choose to store auditing information in a file by setting the **auditStore** global option to `file`. For more information about the **auditStore** global option, see "Global options - detailed description" on page 16.

Each audit log provides audit information for one day, from 00:00:00 UTC to 23:59:59 UTC regardless of the time zone of the local workstation, but the log file is created only when an action is performed or the WebSphere Application Server is started.

The files are called `yyyymmdd`, and are created in the following directories:

`<TWA_home>/TWS/audit/plan`

`<TWA_home>/TWS/audit/database`

Audit entries are logged to a flat text file on individual workstations in the IBM Workload Scheduler network to minimize the risk of audit failure due to network issues. The log formats are the same for both plan and database. The logs consist of a header portion which is the same for all records, an action ID, and a section of data that varies according to the action type. All data is kept in clear text and formatted to be readable and editable from a text editor such as **vi** or **notepad**.

For more information about the details available in the logs, see "Audit log header format" on page 388 and "Audit log body format" on page 388.

Note: For **modify** commands, two entries are made in the log for resources, calendars, parameters, and prompts. The **modify** command is displayed in the log as a combination of the **delete** and **add** commands.

Storing auditing information in the database (auditStore=db)

This storage location is available when you audit information in the database (**enDbAudit** global option). Choose to store auditing information in the database by setting the **auditStore** global option to `db`. For more information about the **auditStore** global option, see "Global options - detailed description" on page 16.

The AUDIT_STORE_RECORDS_V table is created in the IBM Workload Scheduler database.

For more information, see the section about the AUDIT_STORE_RECORDS_V table in *IBM Workload Scheduler: Database Views*.

Storing auditing information both in the database and in a file (auditStore=both)

This storage location is available when you audit information in the database (**enDbAudit** global option). Choose to store auditing information both in the database and in a file by setting the **auditStore** global option to `both`. For more information about the **auditStore** global option, see "Global options - detailed description" on page 16.

For details about how the information is stored, see "Storing auditing information in the database (auditStore=db)" on page 387 and "Storing auditing information in a file (auditStore=file)" on page 387.

Audit log header format

Each log file starts with a header record that contains information about when the log was created and whether it is a plan or database log.

The header record fields are separated by vertical bars (|), as follows:

```
HEADER|<GMT_date>|<GMT_time>|<local_date>|<local_time>|<object_type>| >
<workstation>|<user_ID>|<version>| <level>
```

Log Type
HEADER

GMT Date
The GMT date when the log file was created.

GMT Time
The GMT time when the log file was created.

Local Date
The local date when the log file was created. The local date is defined by the time zone option of the workstation.

Local Time
The local time when the log file was created. The local time is defined by the time zone option of the workstation.

Object Type
DATABASE for a database log file and PLAN for a plan log file.

Workstation Name
The IBM Workload Scheduler workstation name for which this file was created. Each workstation in the IBM Workload Scheduler network creates its own log.

User ID
The IBM Workload Scheduler user ID that created the log file.

Version
The version of the file.

Level The logging level.

Audit log body format

The audit log formats are basically the same for the plan and the database. The log consists of a header portion, an action ID, and data sections that vary with the action type. The data is in clear text format and each data item is separated by a vertical bar (|).

The log file entries are in the following format:

```
<log_type>|<GMT_date>|<GMT_time>|<local_date>|<local_time>|<object_type>| >
<action_type>|<workstation>|<user_ID>|<object_name>|<action_data_fields>
```

The log files contain the following information:

log_type
Displays an eight character value indicating the source of the log record. The following log types are supported:

CONMAN
conman command text

DATABASE
Database action

HEADER
The log file header

MAKESEC
makesec run

PARMS
Parameter command text

PLAN Plan action

RELEASE
release command text

STAGEMAN
stageman run

GMT_date
Displays the GMT date the action was performed. The format is *yyyymmdd* where *yyyy* is the year, *mm* is the month, and *dd* is the day.

GMT_time
Displays the GMT time the action was performed. The format is *hhmmss* where *hh* is the hour, *mm* is the minutes, and *ss* is the seconds.

local_date
Displays the local date the action was performed. The local date is defined by the time zone option of the workstation. The format is *yyyymmdd* where *yyyy* is the year, *mm* is the month, and *dd* is the day.

local_time
Displays the local time the action was performed. The local time is defined by the time zone option of the workstation. The format is *hhmmss* where *hh* is the hour, *mm* is the minutes, and *ss* is the seconds.

object_type
Displays the type of the object that was affected by an action, from the following:

DATABASE
Database definition (for header only)

DBCAL
Database calendar definition

DBDOMAIN
Database domain definition

DBJBSTRM
Database Job Scheduler definition

DBJOB
Database job definition

DBPARM
Database parameter definition

DBPROMPT
Database prompt definition

DBRES
Database resource definition

DBSEC
Database security

DBUSER
Database user definition

DBVARTAB
Database variable table definition

DBWKCLS
Database workstation class definition

DBWKSTN
Database workstation definition

PLAN Plan (for header only)

PLDOMAIN
Plan domain

PLFILE
Plan file

PLJBSTRM
Plan Job Scheduler

PLJOB
Plan job

PLPROMPT
Plan prompt

PLRES
Plan resource

PLWKSTN
Plan workstation

action_type
Displays what action was performed on the object. The appropriate values for this field are dependent on which action is being performed.

For the plan, the <action_type> can be ADD, DELETE, MODIFY, or INSTALL.

For the database, the ADD, DELETE and MODIFY actions are recorded for workstation, workstation classes, domains, users, jobs, job streams, calendars, prompts, resources and parameters in the database.

The <action_type> field also records the installation of a new Security file. When **makesec** is run, IBM Workload Scheduler records it as an INSTALL action for a Security definition object.

LIST and DISPLAY actions for objects are not logged.

For parameters, the command line with its arguments is logged.

workstation
Displays the IBM Workload Scheduler workstation from which the user is performing the action.

user_ID
Displays the logon user who performed the particular action. On Windows operating systems, if the user who installed WebSphere Application Server was a domain user, for Log Types **stageman** and **conman** this field contains the fully qualified user ID *domain\user*.

object_name
Displays the fully qualified name of the object. The format of this field depends on the object type as shown here:

DATABASE
N/A

DBCAL
<*calendar*>

DBDOMAIN
<*domain*>

DBJBSTRM
<*workstation*>#<*job_stream*>

DBJOB
<*workstation*>#<*job*>

DBPARM
<*workstation*>#<*parameter*>

DBPROMPT
<*prompt*>

DBRES
<*workstation*>#<*resource*>

DBSEC
N/A

DBUSER
[<*workstation*>#]<*user*>

DBVARTAB
<*variable_table*>

DBWKCLS
<*workstation_class*>

DBWKSTN
<*workstation*>

PLAN N/A

PLDOMAIN
<*domain*>

PLFILE
<workstation>#<path>(<qualifier>)

PLJBSTRM
<workstation>#<job_stream_instance>

PLJOB
<workstation>#<job_stream_instance>.<job>

PLPROMPT
[*<workstation>*#]*<prompt>*

PLRES
<workstation>#<resource>

PLWKSTN
<workstation>

action_data_fields
Displays the action-specific data fields. The format of this data is dependent on the <action_type> field.

Sample audit log entries:
This is a sample database audit log:

```
HEADER  |20080207|084124|20080207|094124|DATABASE|      |WK1|         | | |Version=A1.0| Level=1

DATABASE|20080207|084124|20080207|094124|DBRES   |ADD   |WK1|operator1| |res=WK1#RESOURCE    |

DATABASE|20080207|100524|20080207|110524|DBWKSTN |MODIFY|WK1|operator1| |ws=TIVOLI10         |

DATABASE|20080207|100525|20080207|110525|DBWKSTN |MODIFY|WK1|operator1| |ws=ASLUTRI1         |

DATABASE|20080207|100525|20080207|110525|DBWKSTN |MODIFY|WK1|operator1| |ws=WK1              |

DATABASE|20080207|100526|20080207|110526|DBDOMAIN|MODIFY|WK1|operator1| |dom=MASTERDM        |

DATABASE|20080207|100610|20080207|110610|DBWKSTN |MODIFY|WK1|operator1| |ws=TIVOLI10         |

DATABASE|20080207|100610|20080207|110610|DBWKSTN |MODIFY|WK1|operator1| |ws=ASLUTRI1         |

DATABASE|20080207|100611|20080207|110611|DBWKSTN |MODIFY|WK1|operator1| |ws=WK1              |

DATABASE|20080207|100611|20080207|110611|DBWKSTN |ADD   |WK1|operator1| |ws=WK2              |

DATABASE|20080207|100612|20080207|110612|DBDOMAIN|MODIFY|WK1|operator1| |dom=MASTERDM        |
```

This is a sample plan audit log:

```
HEADER  |20080207|100758|20080207|110758|PLAN    |       |WK1|admin| |     |Version=A1.0|Level=1

STAGEMAN|20080207|100758|20080207|110758|PLAN    |INSTALL|WK1|admin| |C:\IBM\TWS\oper1\Symphony|
                   AWSBHV030I The new Symphony file is installed.

STAGEMAN|20080207|100758|20080207|110758|PLAN    |INSTALL|WK1|admin| |C:\IBM\TWS\oper1\Sinfonia|
                 AWSBHV036I Multi-workstation Symphony file copied to C:\IBM\TWS\oper1\Sinfonia

STAGEMAN|20080207|100758|20080207|110758|ADITLEVL|MODIFY |WK1|admin| |                         |
                   AWSBHV077I Audit level changing from 0 to 1.

CONMAN  |20080207|100800|20080207|110800|PLWKSTN |MODIFY |   |admin| |WK1                      |
                   continue & start
```

```
CONMAN  |20080207|100941|20080207|110941|PLWKSTN |MODIFY |   |admin| |SLUTRI1                |
                  limit cpu=slutri1;10

PLAN    |20080207|101018|20080207|111018|PLWKSTN |MODIFY |WK1|oper1| |WK1                    |
                  limit cpu=SLUTRI1;20

PLAN    |20080207|101028|20080207|111028|PLDOMAIN|MODIFY |WK1|oper1| |ECCOLO                 |
                  reply ECCOLO;yes
```

A **ResetPlan** command run against the current production plan is stored in the plan audit log file as follows:

```
STAGEMAN|20080207|100758|20080207|110758|PLAN|DELETE|WK1|admin|
     |/home/WK1/schedlog/M200803140127|
                  AWSBHV025I The old Symphony file renamed /home/WK1/schedlog/M200803140127
```

Dynamic workload scheduling audit

Description

When you select the dynamic scheduling capability at installation time, the auditing feature is automatically installed. By default, the auditing feature is disabled.

Auditable events are as follows:

JobDefinitionAuditEvent
: Maintains a track of operations performed on job definitions.

JobLogAuditEvent
: Maintains a track of operations performed on job logs.

JobAuditEvent
: Maintains a track of operations performed on jobs.

ResourceAuditEvent
: Maintains a track of operations performed on resources.

RelationshipAuditEvent
: Maintains a track of operations performed on relationships between resources.

RecoveryActionAuditEvent
: Maintains a track of operations performed on recovery actions.

HistoryDataAuditEvent
: Maintains a track of operations performed on historical data.

To configure the auditing of events, enable the auditing feature and optionally change the default values in the configuration file to define event types to be audited. The configuration file is located in the following path:

`TWA_home\TDWB\config\audit.properties`

Configuring the audit

Configure one or more of the properties in the `audit.properties` file to enable and configure auditing:

audit.enabled
: Specifies whether the auditing feature is enabled or disabled. The default value is false. Supported values are as follows:

false The auditing feature is not enabled.

true The auditing feature is enabled.

onSecurityEnabled
The auditing feature is enabled if global security is enabled on the WebSphere Application Server.

audit.consumer.file.auditFilePrefix
Specifies the file prefix for the auditing log file. The file name is defined using the file prefix plus the _audit*N*.log suffix, where *N* is a progressive number. If you want the date and time of the file creation specified in the file prefix, use the default format: 'tdwb_'yyyy-MM-dd. For instance, using the default prefix 'tdwb_'yyyy-MM-dd generates the tdwb_2010-12-20_audit*N*.log family of files. Note that the text between single quotation marks (') is not processed by the program and remains unchanged. This format creates a different file for each day the auditing feature is enabled. Also, changing the prefix to 'tdwb_'yyyy-MM generates the tdwb_2010-12_audit*N*.log family of files. This format creates a different file for each month the auditing feature is enabled.

You can modify this format as required to create files on a weekly, monthly or yearly basis, depending on your auditing requirements. Depending on the date and time format you choose, the maximum size and number of log files vary. The maximum size and number of log files are defined using the **audit.consumer.file.maxFileSize** and **audit.consumer.file.maxAuditFiles** properties respectively. Use these three parameters to control the size of the audit logs stored. For example, using the default values for these parameters, then every day you will have a maximum of 10 MB x 100 files each day. Once the maximum is reached, the first file created is overwritten. If you want use less space to store audit logs, you can decided to change the maximum number of files or only have files on a monthly basis, by specifying the format for the `audit.consumer.file.auditFilePrefix` property as 'tdwb_'yyyy-MM.

audit.consumer.file.auditFileLocation
Specifies the path where the log files are created. The default path is `/audit`.

audit.consumer.file.maxFileSize
Specifies the maximum size in bytes of the log files. When a file reaches the maximum size, a new log file is created. The default value is 10000000 bytes (10 MB). This is also the highest supported value.

audit.consumer.file.maxAuditFiles
Specifies the maximum number of files with a specific prefix. When all files reach the maximum size and the maximum number of files is exceeded, the oldest file with a specific prefix is overwritten. The default value is 100 files. This is also the highest supported value.

Configuring dynamic audit events

The following table lists the supported actions and properties for each event with the related default values. You can configure these values in the `audit.properties` file.

Table 76. Auditable event properties

Event	Action	Property	Default value
JobDefinitionAuditEvent	create	audit.tdwb.JobDefinitionAuditEvent.create.enabled	true
	delete	audit.tdwb.JobDefinitionAuditEvent.delete.enabled	true
	get	audit.tdwb.JobDefinitionAuditEvent.get.enabled	true
	query	audit.tdwb.JobDefinitionAuditEvent.query.enabled	false
	update	audit.tdwb.JobDefinitionAuditEvent.update.enabled	true
JobLogAuditEvent	get	audit.tdwb.JobLogAuditEvent.get.enabled	true
JobAuditEvent	cancel	audit.tdwb.JobAuditEvent.cancel.enabled	true
	get	audit.tdwb.JobAuditEvent.get.enabled	true
	query	audit.tdwb.JobAuditEvent.query.enabled	false
	submit	audit.tdwb.JobAuditEvent.submit.enabled	true
ResourceAuditEvent	create	audit.tdwb.ResourceAuditEvent.create.enabled	true
	delete	audit.tdwb.ResourceAuditEvent.delete.enabled	true
	query	audit.tdwb.ResourceAuditEvent.query.enabled	false
	resume	audit.tdwb.ResourceAuditEvent.resume.enabled	true
	suspend	audit.tdwb.ResourceAuditEvent.suspend.enabled	true
	update	audit.tdwb.ResourceAuditEvent.update.enabled	true
RelationshipAuditEvent	create	audit.tdwb.RelationshipAuditEvent.create.enabled	true
	delete	audit.tdwb.RelationshipAuditEvent.delete.enabled	true
	query	audit.tdwb.RelationshipAuditEvent.query.enabled	false
RecoveryActionAuditEvent	invoke	audit.tdwb.RecoveryActionAuditEvent.invoke.enabled	true
HistoryDataAuditEvent	move	audit.tdwb.HistoryDataAuditEvent.move.enabled	true

By default, auditing is disabled for query actions, while all the other actions are enabled. If the auditing feature is disabled, all properties are ignored.

Log file specifications

The elements used in the auditing log files are extensions to the Common Base Event (CBE) schema. The types and elements listed below are available in the auditing log files. Supported action types for each element are listed in Table 76.

Action

Represents the action that is being taken. Each auditable event supports a different set of possible actions. See Table 76. The Action type contains the following element:

Table 77. Elements in Action type

Element name	Element description	Always returned in the output
Action	The action type that is being taken on thedynamic workload broker object.	Yes

ObjectInfoList

Represents a list of dynamic workload broker objects. The `ObjectInfoList` type contains the following element:

Table 78. Elements in ObjectInfoList type

Element name	Element description	Always returned in the output
objectInfo	The class of the object being involved in the action	Yes

ObjectInfo

Represents information about a dynamic workload broker object in an `objectInfoList` type or in another `objectInfo` element. The `ObjectInfo` type contains the following elements:

Table 79. Elements in ObjectInfo type

Element name	Element description	Always returned in the output
objectClass	The class of the object being involved in the action.	Yes
objectName	The name of the dynamic workload broker object.	Only if available
objectNamespace	The namespace of the dynamic workload broker object.	Only if available
objectType	The type of the dynamic workload broker object.	Only if available
objectAlias	The alias of the dynamic workload broker object.	Only if available
objectIdentifier	The unique identifier of the dynamic workload broker object.	Only if available
objectRole	The role of the dynamic workload broker object, if any. For instance a Resource can have the source or destination role in a relationship	Only if available
objectSubmitterType	The type of the component which submitted the operation. The component is one of the following: • Dynamic Workload Broker Console • Command line • Dynamic workload broker workstation • Third party utility	Only if available
objectInfo	A child `objectInfo` object. For instance, a relationship is always related to two resources.	Only if available

Outcome

Defines the outcome of a security event. The Outcome type contains the following elements:

Table 80. Elements in Outcome type

Element name	Element description	Always returned in the output
result	The status of the event. This information can be used when filtering the information in the log file.	Yes
failureReason	Additional information on the outcome of the operation.	Yes, if the operation was unsuccessful.

UserInfoList

Represents a list of `userInfo` elements, each representing the list of users in the delegation chain. The `UserInfoList` type contains the following element:

Table 81. Elements in UserInfoList type

Element name	Element description	Always returned in the output
objectInfo	An array of Information about each user in the delegation chain. The first `userInfo` element identifies the user which authenticated first. The last `userInfo` element identifies the user with whose credentials the action is being taken.	Yes

UserInfo

Represents information about a user. Elements of this type return information about the user involved in the operation being audited. The `UserInfo` type contains the following element:

Table 82. Elements in UserInfo type

Element name	Element description	Always returned in the output
UserInfo	The username provided to dynamic workload broker for authentication.	Yes

How to perform queries on log files

Log files can be very long and detailed. When you view your log files with the Log and Trace Analyzer, you can apply one or more queries to filter information in the file and make searches faster. You can use the following queries to filter only the relevant information or you can create your own queries depending on your requirements. The following queries are written in XPath query language.

- To filter all the events generated by a specific user:

  /CommonBaseEvent [extendedDataElements/children[@name='*userInfo*' and values='*username*']]
- To filter all the events related to a specific object class:

  /CommonBaseEvent [extendedDataElements//children[@name='objectClass' and values='Resource]]
- To filter all the events related to a specific object:

//CommonBaseEvent [extendedDataElements//children[@name='objectName' and values='myresource']/../children[@name='objectClass' and values='Resource']]

- To filter all the events related to a specific action:

  /CommonBaseEvent [extendedDataElements[@name='action' and values='uninstall']]

- To filter all the events with SUCCESSFUL outcome:

  /CommonBaseEvent [extendedDataElements/children[@name='result' and values='SUCCESSFUL']]

The following query returns all create actions:

```
/CommonBaseEvent[ extendedDataElements[@name = 'action' and values = 'create']]
```

You can export this query into an XML file as follows:

```
<?xml version="1.0" encoding="UTF-8"?><cbeviewer_configuration>
<logParserSets>
   <logParserSet  description="Parser for CBE log"
                  id="com.ibm.cbeviewer.parsers.cbeLogParserSet"
                  label="Common Base Event log"
                  parentId="com.ibm.cbeviewer.parsers.jdLogParserSet"/>
   <logParserSet  description="Parser for CEI Server"
                  id="com.ibm.cbeviewer.parsers.ceiLogParserSet"
                  label="Common Event Infrastructure server"
                  parentId="com.ibm.cbeviewer.parsers.jdLogParserSet"/>
   <logParserSet description="Other parsers"
                  id="com.ibm.cbeviewer.parsers.otherParsersLogParserSet"
                  label="Other parsers"/>
</logParserSets>
<recent_expressions>
     <xpath name="All Create Events">
     /CommonBaseEvent[ extendedDataElements[@name = 'action' and values = 'create']]
     </xpath>
</recent_expressions></cbeviewer_configuration>
```

The following is a short example of a log file:

```
<CommonBaseEvent
      creationTime="2007-06-06T14:26:23.311Z"
      extensionName="TDWB_JOB_AUDIT_EVENT"
      globalInstanceId="CEFC6DD156CA54D902A1DC1439E6EC4ED0"
      sequenceNumber="1"
      version="1.0.1">
   <extendedDataElements
         name="userInfoList"
         type="noValue">
      <children
            name="userInfo"
            type="string">
         <values>UNAUTHENTICATED</values>
      </children>
   </extendedDataElements>
   <extendedDataElements
         name="action"
         type="string">
      <values>submit</values>
   </extendedDataElements>
   <extendedDataElements
         name="outcome"
         type="noValue">
      <children
            name="result"
            type="string">
```

```
        <values>SUCCESSFUL</values>
      </children>
   </extendedDataElements>
</CommonBaseEvent>
```

Examples

The following examples describe a standard usage of the auditing feature.

In the following example, user `root` successfully retrieves the definition of a job named **MyTestJob** using the `jobstore` command.

```
<CommonBaseEvent
      creationTime="2007-06-21T16:05:19.455Z"
      extensionName="TDWB_JOB_AUDIT_EVENT"
      globalInstanceId="CE8F5E102AE3419AF7A1DC201135463A40"
      sequenceNumber="188"
      version="1.0.1">
   <extendedDataElements
         name="userInfoList"
         type="noValue">
      <children
            name="userInfo"
            type="string">
         <values>root</values>
      </children>
   </extendedDataElements>
   <extendedDataElements
         name="action"
         type="string">
      <values>get</values>
   </extendedDataElements>
   <extendedDataElements
         name="outcome"
         type="noValue">
      <children
            name="result"
            type="string">
         <values>SUCCESSFUL</values>
      </children>
   </extendedDataElements>
   <extendedDataElements
         name="objectInfoList"
         type="noValue">
      <children
            name="objectInfo"
            type="noValue">
         <children
               name="objectClass"
               type="string">
            <values>Job</values>
         </children>
         <children
               name="objectName"
               type="string">
                  <values>MyTestJob</values>
         </children>
         <children
               name="objectIdentifier"
               type="string">
            <values>3ebf6d62-0b83-3270-9b83-83c393e9cbca</values>
         </children>
         <children
               name="objectSubmitterType"
               type="string">
            <values>TDWB CLI</values>
         </children>
      </children>
   </extendedDataElements>
   <extendedDataElements
         name="CommonBaseEventLogRecord:sequenceNumber"
```

```
        type="long">
       <values>80808</values>
   </extendedDataElements>
   <extendedDataElements
        name="CommonBaseEventLogRecord:threadID"
        type="int">
      <values>280</values>
   </extendedDataElements>
   <sourceComponentId
        application="JobManagement"
        component="None"
        componentIdType="Application"
        location="tdws08"
        locationType="Hostname"
        subComponent="None"
        threadId="Default : 84"
        componentType="http://www.ibm.com/namespace/autonomic/Tivoli_componentTypes"/>
   <situation
        categoryName="ReportSituation">
      <situationType
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xsi:type="ReportSituation"
           reasoningScope="INTERNAL"
           reportCategory="SECURITY"/>
   </situation>
</CommonBaseEvent>
```

In the following example, user `testuser` tries deleting a job instance named **MySecondJob** using the appropriate command line. The operation fails because the job was submitted by another user. Deleting jobs submitted by other users requires `Operator` or `Administrator` rights. For more information on access rights, see *IBM Workload Scheduler: Scheduling Workload Dynamically* or *IBM Workload Scheduler: Administration Guide*.

```
<CommonBaseEvent
      creationTime="2007-06-21T16:05:32.746Z"
      extensionName="TDWB_JOB_AUDIT_EVENT"
      globalInstanceId="CE8F5E102AE3419AF7A1DC20113D32BB20"
      sequenceNumber="189"
      version="1.0.1">
   <extendedDataElements
        name="userInfoList"
        type="noValue">
      <children
           name="userInfo"
           type="string">
        <values>testuser</values>
      </children>
   </extendedDataElements>
   <extendedDataElements
        name="action"
        type="string">
      <values>cancel</values>
   </extendedDataElements>
   <extendedDataElements
        name="outcome"
        type="noValue">
      <children
           name="result"
           type="string">
        <values>UNSUCCESSFUL</values>
      </children>
      <children
           name="failureReason"
           type="string">
        <values>userNotAuthorized</values>
      </children>
   </extendedDataElements>
   <extendedDataElements
        name="objectInfoList"
```

```
      type="noValue">
    <children
         name="objectInfo"
         type="noValue">
      <children
           name="objectClass"
           type="string">
        <values>Job</values>
      </children>
      <children
           name="objectName"
           type="string">
        <values>MySecondJob</values>
      </children>
      <children
           name="objectIdentifier"
           type="string">
        <values>a05732c8-c008-3103-afd1-84b567d78de7</values>
      </children>
      <children
           name="objectSubmitterType"
           type="string">
        <values>TDWB CLI</values>
      </children>
    </children>
  </extendedDataElements>
  <extendedDataElements
       name="CommonBaseEventLogRecord:sequenceNumber"
       type="long">
    <values>80964</values>
  </extendedDataElements>
  <extendedDataElements
       name="CommonBaseEventLogRecord:threadID"
       type="int">
    <values>292</values>
  </extendedDataElements>
  <sourceComponentId
       application="JobManagement"
       component="None"
       componentIdType="Application"
       location="tdws08"
       locationType="Hostname"
       subComponent="None"
       threadId="Default : 91"
       componentType="http://www.ibm.com/namespace/autonomic/Tivoli_componentTypes"/>
  <situation
       categoryName="ReportSituation">
    <situationType
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:type="ReportSituation"
         reasoningScope="INTERNAL"
         reportCategory="SECURITY"/>
  </situation>
</CommonBaseEvent>
```

Keeping track of database changes using audit reports

To keep always track of the changes that impact objects stored in the database, you can use the following audit reports, which can be run in batch mode using the command line interface:

General audit report
The report provides information about objects that have been modified in the database. More specifically, it details who made the change, on which objects, and when.

Details report
The report provides further details about the changes implemented. It

specifies who made the change, on which objects, when, and what has been changed. More specifically it shows the object definition before and after the change.

You can run these reports on DB2 and Oracle databases.

A sample business scenario

The administrator of an insurance company needs to keep track of all the changes impacting the insurance policies, conditions and terms of all the customers registered in the company database. To do it, the administrator periodically runs the audit general and details reports.

To satisfy this request, he creates an audit general report that provides details about which TWS objects have been modified in the database, who modified them and on which date. Then, to find out more details about the changes, he also creates an audit details report.

To accomplish his task, he runs the following steps:

1. He customizes the property files related to the audit reports, specifying the format and content of the report output.
2. He schedules jobs to obtain the reports:
   a. The first job generates an audit to be saved locally.
   b. The second job runs a detail report overnight to retrieve more details about the specific changes implemented. The report output is sent using an mail to the analyst. The information collected is used to keep all the insurance branch offices updated with any change and news.
3. The administrator adds the two jobs to a job stream scheduled to run weekly and generates the plan.

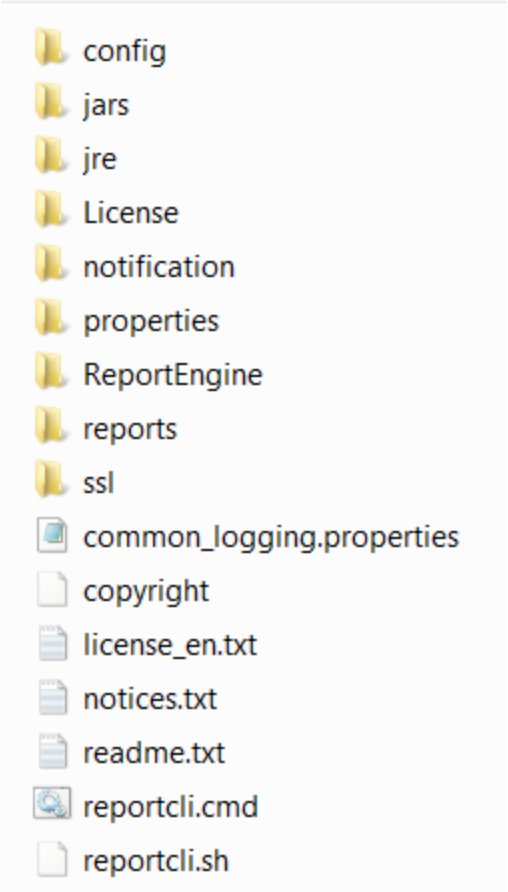
Setting up for command line audit reporting

About this task

Before running these reports you must perform a few setup steps:

1. The software needed to run these reports is contained in a package named `TWSBatchReportCli` included in the IBM Workload Scheduler installation image, in the `TWSBatchReportCli` directory. If you plan to run them from within a scheduled job, extract the package file on one of the operating systems listed in the System Requirements Document at http://www-01.ibm.com/support/docview.wss?rs=672&uid=swg27048859.

   After extracting the package, you obtain the following file structure:



Because the native UNIX tar utility does not support long file names, if you are extracting the files on AIX, Solaris, or HP-UX systems, ensure that the latest GNU version of tar (gtar) is installed to extract the files successfully.

Note:

a. Make sure you run the following commands in the directory where you extracted the files:

On UNIX

```
chmod -R +x *
chown -R username *
```

On Windows

Ensure IBM Workload Scheduler is installed.

```
setown -u username *
```

Where *username* is the IBM Workload Scheduler user that will run the reports.

b. If you plan to schedule jobs that run these reports, the system where you extract the package must be accessible as network file system from a fault-tolerant agent defined in the local scheduling environment.

2. If you use an Oracle database, download the JDBC drivers required by your Oracle server version.
3. Copy the JDBC drivers in the *report_cli_installation_dir*\jars directory and in *report_cli_installation_dir*\ReportEngine\plugins\

`org.eclipse.birt.report.data.oda.jdbc_4.2.1.v20120820\drivers` directory. The report cli automatically discovers the two jar files.

4. Configure the template file `.\config\common.properties` specifying the following information:

   a. If you use an Oracle database, connect to the database where the historical data are stored as follows:

      1) Retrieve the location of the Oracle JDBC drivers. This information is stored in the **com.ibm.tws.webui.oracleJdbcURL** property in the `TWA_home/WAS/TWSprofile/properties/TWSConfig.properties` file. For more information about this file, see "Configuring for an Oracle database" on page 179.

      2) Specify the location of the Oracle JDBC drivers in the **PARAM_DataSourceUrl** property in the `common.properties` file.

      No customization is required if you use DB2.

   b. Set the date and time format, including the time zone. The file `.\config\timezone.txt` contains a list of time zones supported by IBM Workload Scheduler and the information on how to set them. The time zone names are case sensitive.

   c. Make the report output available on the URL specified in **ContextRootUrl** field. This is an example of the configuration settings:

```
######################################################################
# HTTP Server information
######################################################################

 #Specify the context root where the report will be available
 #To leverage this possibility it needs to specify in the report output dir
 #the directory that is referred by your HTTP Server with this contect root

 ContextRootUrl=http://myserver/reportoutput
```

      In this case, en sure that the *output_report_dir* specified when running the reports command points to the same directory specified in the **ContextRootUrl**.

   d. Send the report output using a mail. This is an example of the configuration settings:

```
######################################################################
# Email Server configuration
######################################################################
 PARAM_SendReportByEmail=true

 #SMTP server
 mail.smtp.host=myhost.mydomain.com
 #IMAP provider
 mail.imap.socketFactory.fallback=false
 mail.imap.port=993
 mail.imap.socketFactory.port=993
 #POP3 provider
 mail.pop3.socketFactory.fallback=false
 mail.pop3.port=995
 mail.pop3.socketFactory.port=995

 ######################################################################
 # Email properties
 ######################################################################
 PARAM_EmailFrom=user1@your_company.com
 PARAM_EmailTo=user2@your_company.com,user3@your_company.com
 PARAM_EmailCC=user4@your_company.com
```

```
PARAM_EmailBCC=user5@your_company.com
PARAM_EmailSubject=Test send report by email
PARAM_EmailBody=This is the report attached
```

An explanation of all the customizable fields is contained in the template file.

Running audit reports from the command line

To run audit report on the database, you must first enable the audit feature and configure the audit options described in "Global options - detailed description" on page 16.

The `\reports\templates` directory contains a sample template file for each type of report.

Before running any of these reports make sure you customize the corresponding template file, either `ad.properties` or `ag.properties`.

In that file, named *report_name*`.properties`, you can specify:

- The information to display in the report header.
- How to filter the information to display the expected result.
- The format and content of the report output.

For more information about the specific settings see the explanation provided in the template file beside each field.

After you set up the environment as it is described in "Setting up for command line audit reporting" on page 402, and you configured report template file, use the following syntax to run the report:

reportcli -p *report_name.property*
[**-o** *output_report_dir*]
[**-r** *report_output_name*]
[**-k** key=*value*]
[**-k** key=*value*]
.....

where:

-p *report_name.property*
Specifies the path name to the report template file.

-o *routput_report_dir*
Specifies the output directory for the report output.

-r *report_output_name*
Specifies the name of the report output.

-k key=*value*
Specifies the value of a settings. This value override the corresponding value, if defined, in the `common.properties` file or in the *report_name*`.properties` file.

Examples:

1. In this example the `reportcli.cmd` is run with the default parameter:
   ```
   reportcli.cmd -p D:\ReportCLI\TWSReportCli\reports\templates\ag.properties
   -r audit1
   ```
2. In this example the `reportcli.cmd` is run using the `-k` parameter to override the values set for **PARAM_DateFormat** in the `.\config\common.properties` file:

```
reportcli.cmd -p D:\ReportCLI\TWSReportCli\reports\templates\ag.properties
-r audit2 -k PARAM_DateFormat=short
```

3. In this example the `reportcli.cmd` is run using the `-k` parameter to override he format specified for the report output in the `.properties` file:

```
./reportcli.sh -p /TWSReportCli/REPCLI/reports/templates/ag.properties
-r audit3 -k REPORT_OUTPUT_FORMAT=html -k OutputView=charts
```

Note: If the report is run through a IBM Workload Scheduler job, the output of the command is displayed in the job output.

Collecting job metrics

You can run the following SQL queries on the Workload Scheduler data base to retrieve the number of jobs run by IBM Workload Scheduler over a period of time. One query determines the number of jobs run by specific workstations, while the other query determines the number of jobs run on the entire IBM Workload Scheduler domain. You can run the queries from the command line interface of your database or you can add them in the Dynamic Workload Console to create your custom SQL reports, as described in section creating a task to create custom SQL reports in *Dynamic Workload Console User's Guide*.

Job metrics queries for DB2

Use the following SQL query to find the number of jobs run on specific workstations:

```
SELECT year(job_run_date_time) AS Year, month(job_run_date_time) AS Month,
cast (count(job_run_date_time) AS INT) AS JobNbr FROM mdl.job_history_v
WHERE workstation_name IN ('WKS_1', 'WKS_2', 'WKS_N') or
(workstation_name = '-' and JOB_STREAM_WKS_NAME_IN_RUN in('WKS_1', 'WKS_2', 'WKS_N') )
GROUP BY year(job_run_date_time), month(job_run_date_time)
```

where 'WKS_1', 'WKS_2', 'WKS_N' are the names of the workstations that ran the jobs you want counted.

Use the following SQL query to find the number of jobs run on the entire IBM Workload Scheduler domain:

```
SELECT year(job_run_date_time) AS Year, month(job_run_date_time) AS Month,
cast (count(job_run_date_time) AS INT) AS JobNbr FROM mdl.job_history_v
GROUP BY year(job_run_date_time), month(job_run_date_time)
```

Job metrics queries for DB2 for zOS

Use the following SQL query to find the number of jobs run on specific workstations:

```
SELECT year(job_run_date_time) AS Year, month(job_run_date_time) AS Month,
cast (count(job_run_date_time) AS INT) AS JobNbr FROM mdl.job_history_v
WHERE workstation_name IN ('WKS_1', 'WKS_2', 'WKS_N')
GROUP BY year(job_run_date_time), month(job_run_date_time)
```

where 'WKS_1', 'WKS_2', 'WKS_N' are the names of the workstations that ran the jobs you want counted.

Use the following SQL query to find the number of jobs run on the entire IBM Workload Scheduler domain:

```
SELECT year(job_run_date_time) AS Year, month(job_run_date_time) AS Month,
cast (count(job_run_date_time) AS INT) AS JobNbr FROM mdl.job_history_v
GROUP BY year(job_run_date_time), month(job_run_date_time)
```

Job metrics queries for Oracle database

Use the following SQL query to find the number of jobs run on specific workstations:

```
SELECT EXTRACT(year FROM job_run_date_time) AS Year,
EXTRACT(month FROM job_run_date_time) AS Month,
cast (count(job_run_date_time) AS INT) AS JobNbr FROM job_history_v
WHERE workstation_name IN ('WKS_1', 'WKS_2', 'WKS_N')
or (workstation_name = '-' and JOB_STREAM_WKS_NAME_IN_RUN in('WKS_1', 'WKS_2', 'WKS_N'))
GROUP BY EXTRACT(year FROM job_run_date_time), EXTRACT(month FROM job_run_date_time);
```

where 'WKS_1', 'WKS_2', 'WKS_N' are the names of the workstations that ran the jobs you want counted.

Use the following SQL query to find the number of jobs run on the entire IBM Workload Scheduler domain:

```
SELECT EXTRACT(year FROM job_run_date_time) AS Year,
EXTRACT(month FROM job_run_date_time) AS Month,
cast (count(job_run_date_time) AS INT) AS JobNbr FROM job_history_v
GROUP BY EXTRACT(year FROM job_run_date_time), EXTRACT(month FROM job_run_date_time);
```

Chapter 9. Administrative tasks

This chapter describes how to perform some specific administrative tasks on IBM Workload Scheduler, as follows:

The tasks

"Application server - encrypting the profile properties files" on page 452
Several of the application server configuration files contain passwords. To avoid that these remain in the files in plain text, run a utility to encrypt them.

"Application server - updating the Windows services after modifications" on page 452
On Windows, after changing certain data you must also update the Windows service that runs the embedded WebSphere Application Server.

"Application server - updating the SOAP properties after changing the WebSphere Application Server user or its password" on page 453
On UNIX or Linux operating systems, if you have changed the user ID or the password of the WebSphere Application Server administration user either for IBM Workload Scheduler or the Dynamic Workload Console, you must also update the SOAP client properties.

"Application server - configuration files backup and restore" on page 454
The application server configuration manages the data source and security aspects of your IBM Workload Scheduler environment. The files should be regularly backed up and when necessary can be restored.

"Application server - changing the host name or TCP/IP ports" on page 455
If you need to change the host or ports used by the application server, follow the correct procedure.

"Application server - changing the trace properties" on page 458
The application server has a trace facility. This section describes how to increase the trace level to obtain more information for troubleshooting, and how to reduce the level to improve performance.

Changing the application server properties

Several of the above tasks require you to run a common procedure whereby you:

1. Run a utility that displays a set of current application server properties and saves them to a file
2. Edit the file to change the properties
3. Run another procedure to update the application server with the changed properties

This procedure is fully described in "Application server - using the utilities that change the properties" on page 459

Application server utilities reference information
Some reference information on the application server utilities is also provided in "Application server utilities" on page 461. For further reading, see *IBM Redbooks®: WebSphere Application Server V6 System Management & Configuration Handbook*.

Changing a domain manager or dynamic domain manager

About this task

A domain manager or dynamic domain manager might need to be changed because you want it to run on a different workstation, or it might be forced on you as the result of network linking problems or the failure of the domain manager or dynamic domain manager workstation itself. This section, and its subsections, describes how to prepare for and use a backup domain manager or dynamic domain manager. However, if the domain manager to be changed is a master domain manager, there are some specific additional steps to perform; see "Changing a master domain manager" on page 415.

Running without a domain manager has the following effects:

- Agents and subordinate domain managers cannot resolve inter-workstation dependencies, because activity records broadcast by the master domain manager are not being received.
- The upward flow of events is interrupted. This impacts events that report the status of jobs, job streams and dependencies defined on workstations in the IBM Workload Scheduler network hierarchy under the failed domain manager.
- Standard agents that are hosted by the failed domain manager cannot perform any processing, since they depend on the domain manager for all scheduling and job launching.

If the problem is expected to be of short duration, you can wait for the problem to be resolved and IBM Workload Scheduler will recover on its own, as described in the *IBM Workload Scheduler: Troubleshooting Guide* in the section about network linking problems. If you are uncertain about the duration, or if you want to restore normal agent operation, you must switch to a backup, as described in the following sections.

Choosing a backup domain manager or backup dynamic domain manager

Being prepared for network problems makes recovery easier. Set up a backup domain manager or backup dynamic domain manager respectively for each domain manager or dynamic domain manager in your network to more easily ensure that IBM Workload Scheduler peak job scheduling loads are met. Choose any fault-tolerant agent in the domain to be a backup domain manager or backup dynamic domain manager.

Setting up a backup domain manager

Ensure that the *FullStatus* mode is selected in the backup domain manager or backup dynamic domain manager workstation definition.

Also ensure that the backup domain manager is synchronized with respect to time with the domain manager. The most secure way is to use a Network Time Protocol Server to control the time on both systems, with the same repeat interval.

Network security

Network security is enforced using IP address validation. As a consequence, workstation linking (*autolink* option or **link** command) might fail if an agent has

an old Symphony file that does not contain the new domain manager. If a connection fails, remove the old Symphony file on the agent and retry the connection.

Switching a domain manager

Use one of these procedures when you have a short-term loss of a domain manager.

Using the command line

See the procedure described under the **switchmgr** command in the *IBM Workload Scheduler: User's Guide and Reference*.

Using the Dynamic Workload Console

1. Launch the Dynamic Workload Console
2. Connect to the engine of the current domain manager
3. From the navigation bar, click **System Status and Health** > **Environment Monitoring** > **Monitor Workstations**.
4. Select a task to monitor workstations.
5. Select the workstation you want to become the domain manager
6. From the table containing the list of workstations, select a workstation and click **More Actions** > **Become Master Domain Manager**.
7. Click **OK**.

Domain managers remain switched until you perform another switch manager operation, or run **JnextPlan**. To return to the original domain manager without running **JnextPlan**, repeat this procedure.

Complete procedure for switching a domain manager

This section summarizes the steps required to replace a running domain manager with its backup and to complete the procedure by restoring the original domain manager to its function. Follow these steps to make sure that no overlapping problems arise with obsolete versions of the Symphony file. The steps are documented for four scenarios:

Planned outage

The domain manager is replaced with its backup for planned maintenance work (for example, an upgrade of the operating system).

Unplanned outage

The domain manager is replaced with its backup because of an unexpected failure or malfunction.

Short-term

The domain manager is expected to return to service before the next new production period turnover (run of the JnextPlan job).

Long-term

The domain manager is not expected to return to service before the next new production period turnover (run of the JnextPlan job).

Table 83. Complete procedure for switching a domain manager in case of a planned outage.

Planned outage	
Short-term	**Long-term**

Table 83. Complete procedure for switching a domain manager in case of a planned outage. (continued)

Planned outage	
1. Switch the domain manager to a backup workstation. Use either the `conman switchmgr` command or the Dynamic Workload Console. For more information, see the `switchmgr` command in *IBM Workload Scheduler: User's Guide and Reference*. Check that the message boxes for the domain manager undergoing maintenance are large enough not to fill up before it is restored. Increase their size if necessary.	1 Switch the domain manager to a backup workstation. Use either the `conman switchmgr` command or the Dynamic Workload Console. For more information, see the `switchmgr` command in *IBM Workload Scheduler: User's Guide and Reference*. Check that the message boxes for the domain manager undergoing maintenance are large enough not to fill up before it is restored. Increase their size if necessary.
2. Shut down IBM Workload Scheduler processing on the domain manager undergoing maintenance.	2. Shut down IBM Workload Scheduler processing on the original domain manager undergoing maintenance.
3. In the IBM Workload Scheduler database assign the role of domain manager to the backup workstation.	3. In the IBM Workload Scheduler database assign the role of domain manager to the backup workstation.
	4. Set the workstation running the original domain manager to `ignore`, using either the `composer cpuname` command or the Dynamic Workload Console.
When ready to restore the ownership of the domain to the original domain manager:	When ready to restore the ownership of the domain to the original domain manager:
4. Reassign ownership of the domain to the original domain manager in the IBM Workload Scheduler database.	5. Remove the `ignore` flag from the workstation running the original domain manager.
5. Switch from the backup workstation to the domain manager using one of the methods indicated in step 1.	6. Reassign ownership of the domain to the original domain manager in the IBM Workload Scheduler database. Optionally, remove in the original domain manager the `conman start` command from the init procedure and delete any existing copies of the Symphony, Sinfonia, and message box files. This step is recommended to avoid that any outdated symphony present in the computer is automatically triggered at the first startup. You can add `conman start` again later.
6. Link the domain manager from the master to download a fresh version of the Symphony file.	
	7. Switch from the backup workstation to the domain manager using one of the methods indicated in step 1.
	8. Link the domain manager from the master to download a fresh version of the Symphony file.

Table 84. Complete procedure for switching a domain manager after an unplanned outage.

Unplanned outage	
Short-term	**Long-term**

Table 84. Complete procedure for switching a domain manager after an unplanned outage. (continued)

Unplanned outage	
1. Switch the domain manager to a backup workstation. Use either the `conman switchmgr` command or the Dynamic Workload Console. For more information, see the `switchmgr` command in *IBM Workload Scheduler: User's Guide and Reference*. Check that the message boxes for the failing domain manager are large enough not to fill up before it is restored. Increase their size if necessary.	1 Switch the domain manager to a backup workstation. Use either the `conman switchmgr` command or the Dynamic Workload Console. For more information, see the `switchmgr` command in *IBM Workload Scheduler: User's Guide and Reference*. Check that the message boxes for the failing domain manager are large enough not to fill up before it is restored. Increase their size if necessary.
3. In the IBM Workload Scheduler database assign the role of domain manager to the backup workstation.	3. In the IBM Workload Scheduler database assign the role of domain manager to the backup workstation.
	4. Set the workstation running the failing domain manager to `ignore`, using either the `composer cpuname` command or the Dynamic Workload Console.
When ready to restore the ownership of the domain to the original domain manager:	When ready to restore the ownership of the domain to the original domain manager:
4. Reassign ownership of the domain to the original domain manager in the IBM Workload Scheduler database. Optionally, remove in the original domain manager the `conman start` command from the init procedure and delete any existing copies of the Symphony, Sinfonia, and message box files. This step is recommended to avoid that any outdated symphony present in the computer is automatically triggered at the first startup. You can add `conman start` again later. For an "unplanned outage", FTA needs a new Symphony file, on the current master domain manager (previous backup master domain manager) do the following: 1. Verify that it is linked to all agents except the old master domain manager 2. Shut down all IBM Workload Scheduler processes (unlink from all agents). 3. Rename Sinfonia as Sinfonia.orig 4. Copy Symphony to Sinfonia.orig You now have identical Symphony and Sinfonia files.	5. Remove the `ignore` flag from the workstation running the original domain manager.

Table 84. Complete procedure for switching a domain manager after an unplanned outage. (continued)

Unplanned outage	
5. Switch from the backup workstation to the domain manager using one of the methods indicated in step 1.	6. Reassign ownership of the domain to the original domain manager in the IBM Workload Scheduler database.
6. Link the domain manager from the master to download a fresh version of the Symphony file.	Optionally, remove in the original domain manager the `conman start` command from the init procedure and delete any existing copies of the Symphony, Sinfonia, and message box files. This step is recommended to avoid that any outdated symphony present in the computer is automatically triggered at the first startup. You can add `conman start` again later.
	7. Switch from the backup workstation to the domain manager using one of the methods indicated in step 1.
	8. Link the domain manager from the master to download a fresh version of the Symphony file.

Switching a dynamic domain manager

Use the procedure described in "Switching a master domain manager or dynamic domain manager" on page 418 when you have a short-term loss of a dynamic domain manager. The configuration information defined in the dynamic workload broker installed with the dynamic domain manager is automatically saved in the IBM Workload Scheduler database. When you switch to the backup dynamic domain manager, this information is automatically applied to the backup dynamic domain manager.

Changing a master domain manager

About this task

If you lose or want to plan to change a master domain manager, all the comments in the section "Changing a domain manager or dynamic domain manager" on page 411 apply, but in addition consider the following:

Choosing a workstation for backup master domain manager

Since you must transfer files between the master domain manager and its backup, the workstations must have compatible operating systems. Do not combine UNIX with Windows workstations, and in UNIX, do not combine big-endian workstations (HP-UX, Solaris, and AIX) with little-endian workstations (most Intel-based operating systems, including Windows and Linux).

See the *IBM Workload Scheduler: Planning and Installation Guide* for details of the prerequisite requirements of a backup master domain manager.

Promoting an agent to backup master domain manager

It is the normal process to install a backup master domain manager when you set up your scheduling network. However, if you have not done so, and decide later that you need a backup master domain manager, you have two options:

- Install a backup master domain manager on a system that is not currently in the workload scheduling network. Follow the instructions in IBM Workload Scheduler: Planning and Installation
- Promote an agent to backup master domain manager. This option is time-consuming and requires you to interrupt your workload scheduling activities, but if you want to do it, follow the procedure described in this section

You *cannot* promote an agent to backup master domain manager, using a command or procedure that allows continuity of workload scheduling activities.

Instead, if you need to change an agent workstation to become the backup master domain manager, you must interrupt the workload scheduling activities. The procedure is as follows:

1. Check that the workstation satisfies the prerequisites for a backup master domain manager
2. If it does, stop and disable all workload scheduling operations on the workstation
3. Uninstall the agent, following the instructions in IBM Workload Scheduler: Planning and Installation.
4. Install the backup master domain manager on the system where the agent was installed, following the instructions in IBM Workload Scheduler: Planning and Installation.
5. Ensure that the database entry for the workstation is correct for a backup master domain manager. See the IBM Workload Scheduler User's Guide and Reference for information about the workstation definition
6. Define and start any workload scheduling operations you require on the workstation in its new role.

Setting up a backup master domain manager

Ensure that the master domain manager and the backup master domain manager have *FullStatus* turned on in the workstation definition. This is important if you need to resort to long-term recovery, where the backup master domain manager generates a `Symphony` file (runs JnextPlan). If *FullStatus* is not turned on, the former master domain manager shows up as a regular fault-tolerant agent after the first occurrence of JnextPlan. During normal operations, the JnextPlan job automatically turns on the *FullStatus* flag for the master domain manager, if it is not already turned on. When the new master domain manager runs JnextPlan, it does not recognize the former master domain manager as a backup master domain manager unless the flag is enabled. The former master domain manager does not have an accurate `Symphony` file when the time comes to switch back.

Also ensure that the backup master domain manager is synchronized with respect to time with the master domain manager. The securest way is to use a Network Time Protocol Server to control the time on both systems, with the same repeat interval.

Copying files to use on the backup master domain manager

To back up the important master domain manager files to the backup master domain manager, use the following procedure:

1. Copy the `Security` file on the master domain manager to the *<TWA_home>*`/TWS` directory on the backup master domain manager. Add a suffix to the file so that it does not overwrite the backup master domain manager's own Security file, for example, `Security_from_MDM`.

2. Copy all files in the `TWA_home/TWS/mozart` directory.
3. Copy the `localopts` file (see "Setting local options" on page 34 for the location). Add a suffix to the file so that it does not overwrite the backup master domain manager's own `localopts` file; for example, `localopts_from_MDM`.

This procedure must be performed each production period, or whenever there are significant changes to any objects. It can be incorporated into a script.

In addition to these required files, you might also want to copy the following:

- Any scripts you might have written.
- Archived Symphony files, for reference.
- Log files, for reference.

Note: Another approach could be to place all of the above files on a separately mountable file system, that could easily be unmounted from the master domain manager and mounted on the backup master domain manager in the event of need. You would almost certainly want to backup these files in addition, to protect against loss of the separately mountable file system.

To prevent the loss of messages caused by a master domain manager error, you can use the fault-tolerant switch-manager facility.

Switching a master domain manager

Use the procedure described in "Switching a domain manager" on page 412 when you have a short-term loss of a master domain manager.

Master domain managers remain switched until you perform another switch manager operation. To return to the original master domain manager, repeat this procedure before the next production period turnover, unless you do not expect the master domain manager to be available for the next production period turnover (final Job Scheduler and JnextPlan job). In this case, use the procedure in the following section.

Extended loss or permanent change of master domain manager

Use the following procedure to switch to the backup if the original master domain manager is not expected to return to service before the next new production period turnover (final Job Scheduler and JnextPlan job). For UNIX, use forward slashes in path names.

1. Use the conman **stop** function to stop IBM Workload Scheduler on the master domain manager and its backup.
2. If you copied the `Security` file from the master domain manager to the backup master domain manager *with a suffix*, now delete the backup master domain manager's own `Security` file and rename the `Security` file with the suffix as just `Security`.
3. If you copied the `localopts` file from the master domain manager to the backup master domain manager *with a suffix*, now merge the backup master domain manager's own `localopts` file with the `localopts` file from the master domain manager. Look at each property in turn and determine which version you want to keep on what is going to be your new master domain manager. For example, the property *thiscpu* needs to be the one from the backup master

domain manager, but the options for controlling how the processes run can be taken from the master domain manager.

4. On the backup master domain manager cancel the *final* Job Scheduler in the Symphony file (it refers to the next production period's JnextPlan on the old master domain manager).
5. On the backup master domain manager, use composer to modify any important job streams that run on the master domain manager, in particular the *final* Job Scheduler. For each of these, change the workstation name to the name of the backup.
6. Change the workstation definition of the master domain manager from *manager* to *fault-tolerant agent*.
7. Change the workstation definition of the backup master domain manager from *fault-tolerant agent* to *manager*.

   Note: These two steps must be done in the order given, as the system will not allow you to have two *managers* at the same time.
8. On the backup master domain manager, edit the `<TWA_home>/TWS/mozart/globalopts` file and change the *master* option to the name of the backup master domain manager workstation (this is used mainly for reports production)
9. Use the conman **switchmgr** function to switch to the backup master domain manager. See "Switching a domain manager" on page 412.
10. Submit a new *final* Job Scheduler to the new master domain manager (old backup master domain manager).
11. Run **JnextPlan -for 0000** on the new master domain manager to generate the new `Symphony` file.
12. Remember to log on to the backup master domain manager when opening the Dynamic Workload Console, first defining a new engine to access it.
13. If the old master domain manager has failed or is being replaced, you can now delete its workstation definition and remove it from the network.

Switching a master domain manager or dynamic domain manager

Switching a master domain manager or dynamic domain manager affects the running dynamic workload broker server.

The installation of a master domain manager or dynamic domain manager and of its backup workstations includes also the installation of a dynamic workload broker server.

Before you switch your master domain manager or dynamic domain manager to a backup workstation, you must stop the dynamic workload broker server. After the switch completed, you must start the dynamic workload broker server on the new master domain manager or dynamic domain manager. The process is not automatic and you must ensure that you avoid having two concurrently active servers.

If you have to switch the master domain manager or dynamic domain manager because the system running the current workstation failed, make sure that also the dynamic workload broker server is down. In this case, you need only to start the new dynamic workload broker server after switching the master.

If you switch the master domain manager or dynamic domain manager for any other reason than a system failure, and you are switching to a backup workstation while the system running the current master domain manager or dynamic domain manager is up, you risk having two concurrently active servers. To avoid this stop the current dynamic workload broker server before you switch, and start the new instance after the backup workstation takes over.

Here is the procedure to follow every time you switch the master domain manager or dynamic domain manager if you run dynamic scheduling in your network:

1. If the dynamic workload broker server on the current master domain manager or dynamic domain manager is still running, stop it. Use wastool **stopBrokerApplication.sh** on UNIX and Linux or **stopBrokerApplication.bat** on Windows as follows:

   ```
   stopBrokerApplication [-user username -password password] [-port portnumber]
   ```

   where,

   username **and** *password*
   The credentials used at installation. The user and password are optional. By default, the script looks for the credentials in the `soap.client.props` file located in the properties directory of the WebSphere Application Server profile.

   portnumber
   The parameter is optional. If it is not specified, the default is used.

2. Switch the master domain manager or dynamic domain manager to a backup workstation. Use either the `conman switchmgr` command or the Dynamic Workload Console. For more information, see the section about the `switchmgr` command in *IBM Workload Scheduler: User's Guide and Reference*.

3. Start the dynamic workload broker server running with the new workstation. Use wastool **startBrokerApplication.sh** on UNIX and Linux or **startBrokerApplication.bat** on Windows as follows:

   ```
   startBrokerApplication [-user username -password password] [-port portnumber]
   ```

   where,

   username **and** *password*
   The credentials used at installation. The user and password are optional. By default, the script looks for the credentials in the `soap.client.props` file located in the properties directory of the WebSphere Application Server profile.

   portnumber
   The parameter is optional. If it is not specified, the default is used.

4. Link the dynamic workload broker server running with the new workstation running the following command:

   ```
   conman link broker_workstation_name
   ```

Changing key IBM Workload Scheduler passwords

About this task

When you change passwords for key users in your IBM Workload Scheduler environment, there are various operations to perform, depending on which user's password is being changed, the type of operating system on which it is deployed, and the type of IBM Workload Scheduler node where the password is being

changed. You can perform these operations manually, or you can use the **changePassword** script described in "Using the changePassword script" on page 427 to accomplish the necessary operations automatically.

If you decide to proceed manually, the following pages describe what you have to do if the passwords of any of the following users change:

IBM Workload Scheduler instance owner
The <*TWS_user*> (the instance owner) of a IBM Workload Scheduler component (on Windows only).

WebSphere Application Server user
The WebSphere Application Server user (as identified by the WebSphere Application Server tools) which authenticates the <*TWS_user*> being used by IBM Workload Scheduler components.

The database user (J2C) of a IBM Workload Scheduler component:

DB2 If you are using a DB2 database, this is the user ID used to access DB2.

Note: This is different according to whether you have the server or the client installed:

DB2 Server installed
The DB2 administration user (local) is used.

DB2 Client installed
The IBM Workload Scheduler DB2 user on the remote server is used.

Oracle If you are using an Oracle database, the Oracle schema owner user.

Note: The Oracle schema owner is not an operating system ID. Even if it has the same value as an operating system ID on the same computer, it is completely separate, and the passwords are changed separately.

Streamlogon user
The streamlogon user of any job run in the IBM Workload Scheduler environment (jobs running on Windows only)

For all other users of IBM Workload Scheduler, no action is required if their passwords change.

Before changing any passwords, you must first change the password at the operating system level using native commands, as follows:

On UNIX operating systems
use the **passwd** command.

On Windows operating systems
use the **net user** command.

If you use special characters in the password, ensure you use a "\" (backslash) before the special character. The following rules apply:

- **On Windows operating systems:**
- Passwords for users can include any alphanumeric characters and
- ()!?=^*/~[]$_+;:.,@`-#.

On UNIX and LINUX systems:

Passwords for users can include any alphanumeric characters and ()!?=*~_+.-.

If you use the **changePassword** script, the password changes and corresponding operations are performed automatically. For detailed information about the script, refer to "Using the changePassword script" on page 427. If you decide to proceed manually, consult Table 85 to determine if a change of password requires actions to be taken for a role on the different IBM Workload Scheduler components. Look up the role and the component and determine from the corresponding table cell where the changes must be made:

- If the cell contains a "✓", make the change on the system where the indicated component is running
- If the cell contains "MDM", make the change on the master domain manager to which the component belongs

Table 85. If and where password changes are required

Role	MDM	BKM	FTA	FTA + CONN
IBM Workload Scheduler instance owner (Windows)	✓	✓	✓	✓
WebSphere Application Server user	✓	✓		✓
Database user	✓	✓		
Streamlogon user (Windows)	✓	✓	MDM	MDM

For example, if you are the TWS_user (the instance owner) of a fault-tolerant agent, you need to implement the password change on the system where the fault-tolerant agent is installed, but if you are also the streamlogon user of jobs running on that system, the changes required for the new password must be applied at the master domain manager to which the fault-tolerant agent belongs.

If you are not certain which user role you are playing, consult "Determining the role of the user whose password has changed."

When you have determined what role you are playing, determine if you need to take any actions, and if so, where, by consulting "Determining the actions to take" on page 422.

Determining the role of the user whose password has changed

About this task

Use the following procedure to determine which role or roles the user whose password has changed is playing.

Attention: A user might have more than one role, in which case you must follow more than one procedure to change the password.

1. Check if the user is the IBM Workload Scheduler instance owner:

Windows

Check if the user whose password is to be changed is the user that owns the *IBM Workload Scheduler for <TWS_user>* service.

UNIX Run the following command:

```
ps -ef | grep netman
```

If the user whose password has changed matches the user ID revealed in this step, then the user is the *IBM Workload Scheduler instance owner*.

2. Check if the user is the WebSphere Application Server user or the database user, or both:

1. Log on to the computer where IBM Workload Scheduler is installed as the following user:

   UNIX root

   Windows
   Any user in the *Administrators* group.
2. Access the directory: *<TWA_home>*/wastools
3. From that same directory run the following script:

   UNIX **showSecurityProperties.sh > <output_file.txt>**

   Windows
   showSecurityProperties.bat > <output_file.txt>

   Note: This command might display a message from the application server (WASX7357I:) in the output file. You can ignore this message.
4. Open *<output_file.txt>* in a text editor.
5. Run the showSecurityProperties script to check the ServerID associated with the value of the activeUserRegistry key. If the user whose password has changed is the same as the value of the ServerID listed in the Federated Repository Panel, then the user is the *WebSphere Application Server user*.
6. Check the value of the key j2cUserid . If the user whose password is to be changed matches this key, the user is the *database user*.

   Note: If the user is the Oracle schema owner, the password must also be changed within Oracle (see the Oracle documentation).

3. Check if the user is a streamlogon user
Using **composer** or the Dynamic Workload Console, check if the user is identified as a user. If so, the user is a *streamlogon user*.

When you have determined which roles the user plays, see Table 85 on page 421 to determine if and where the password change must be implemented, and then "Determining the actions to take."

Determining the actions to take

Consult Table 86 to determine which actions you need to perform for a change of password:

Table 86. Password change actions

	TWS instance owner (Windows only)	WebSphere Application Server user	Database user	Streamlogon user (Windows only)
"Action 1 - change the WebSphere Application Server user ID password" on page 423		✓ (1 on page 423)		

Table 86. Password change actions (continued)

	TWS instance owner (Windows only)	WebSphere Application Server user	Database user	Streamlogon user (Windows only)
"Action 2 - change password used by command-line clients to access the master domain manager" on page 424		✓		
"Action 3 - change password used by fault-tolerant agent systems to access the master domain manager (for conman)" on page 425		✓		
"Action 4 - update the engine connection parameters in the GUIs" on page 425		✓		
"Action 5 - change the j2c user ID password" on page 425			✓ (1)	
"Action 6 - update SOAP properties" on page 426		✓		
The following step only applies to passwords of users on Windows computers				
"Action 7 - Windows - update Windows services" on page 427	✓	✓		
"Action 8 - change the IBM Workload Scheduler user definition" on page 427				✓

Note:

1. If the user is both the WebSphere Application Server user *and* the database user, the changes made by running **changeSecurityProperties** can be performed as one action, modifying both passwords with the same value.

Action 1 - change the WebSphere Application Server user ID password

About this task

Use the **changeSecurityProperties** utility to change the WebSphere Application Server user ID password.

The procedure requires you to create a text file of the current security properties, edit the file, stop the application server, run the utility and restart the application server.

Note: You might have already created the text file while determining your role (see "Determining the role of the user whose password has changed" on page 421).

Find information about how to do this as follows:

- "Application server - using the utilities that change the properties" on page 459 gives a generic description of the procedure for making any change to the WebSphere Application Server properties
- "Changing the security settings" on page 443 gives reference information about the utility
- When editing the text file of the current security properties, locate either `LocalOSServerpassword` or `LDAPPassword`, depending on the type of

authentication you are using (see "Determining the role of the user whose password has changed" on page 421), and change the password to the new value, in plain text.

Note:

1. If the user is both the WebSphere Application Server user *and* the database user, you can change the properties for both in the same action. See "Action 5 - change the j2c user ID password" on page 425. for details of the property to change.
2. The **changeSecurityProperties** utility might display a message from the application server (WASX7357I:). You can ignore this message.
3. When you supply a password in a text file for **changeSecurityProperties**, there is a small security exposure. When you enter a password in the file, the password is entered in clear (unencrypted). After you have run **changeSecurityProperties**, the password remains in clear in the text file you have edited, but if you run **showSecurityProperties** the password is output encrypted. Thus, your potential security exposure is limited to the time from when you entered the password in the text file until when you manually deleted the text file after using **changeSecurityProperties**.

   Attention: if you subsequently want to change *other* parameters and do *not* want to change any passwords, you must do one of the following before running **changeSecurityProperties**:

   - Resupply the passwords in clear
   - Comment the password properties
   - Delete the password properties

   This is to avoid that the row of asterisks is applied as the password.

Action 2 - change password used by command-line clients to access the master domain manager

About this task

If you have changed the password of the WebSphere Application Server user that command-line clients use to connect to the master domain manager, the connection parameters must be updated.

Follow this procedure:

1. Identify all systems that have a command line client remote connection defined with the master domain manager
2. On these workstations, open the user options files (one for each user). The default file name is *<User_home>*/.TWS/useropts, but if you have more than one instance of IBM Workload Scheduler on a system, you might have implemented separate user options files to make separate connections, in which case consult the *useropts* key in the localopts file on each instance to determine the name of the specific useropts file for that instance.
3. For each file, locate the password key (encrypted) and change its value to that of the new password in plain text, enclosed in double quotation marks. The password is saved in clear, but will be encrypted at first logon of the User ID.
4. Save the files.
5. Check if the following file exists: *<Root_home>*/.TWS/useropts. If it does, change the password in the same way.

Action 3 - change password used by fault-tolerant agent systems to access the master domain manager (for conman)

About this task

If you have changed the password of the WebSphere Application Server user that is used by fault-tolerant agents with an HTTP or HTTPS connection defined in the local options that points to the master domain manager, the connection parameters must be updated.

Follow this procedure:

1. Identify all fault-tolerant agents with an HTTP or HTTPS connection defined in the local options that points to the master domain manager.
2. On these workstations, open the user options file *<Root_home>*/.TWS/useropts
3. Locate the password key (encrypted) and change its value to that of the new password in plain text, enclosed in double quotation marks. The password is saved in clear, but will be encrypted at first logon of the User ID.
4. Save the file.

Action 4 - update the engine connection parameters in the GUIs

About this task

If you have changed the password of the WebSphere Application Server user that is used by the Dynamic Workload Console to connect to the distributed engine, the engine connection parameters must be updated, as follows:

1. On each instance of the Dynamic Workload Console locate the page where you modify the distributed engine connection parameters
2. Change the password and submit the page.

Action 5 - change the j2c user ID password

About this task

Use the **changeSecurityProperties** utility to change the j2c database user ID password.

The procedure requires you to create a text file of the current security properties, edit the file, stop the application server, run the utility, and restart the application server.

Note: You might have already created the text file while determining your role (for more information, see "Determining the role of the user whose password has changed" on page 421).

For detailed information, see the following sections:

- "Application server - using the utilities that change the properties" on page 459 provides a generic description of the procedure for changing the WebSphere Application Server properties.
- "Changing the security settings" on page 443 provides reference information about the **changeSecurityProperties** utility.
- When editing the text file of the current security properties, locate the j2cPassword file and change the password to the new value, in plain text.

Note:

1. If the user is both the WebSphere Application Server user *and* the database user, you can change the properties for both in the same action. For details about the properties to change, see "Action 1 - change the WebSphere Application Server user ID password" on page 423.
2. The **changeSecurityProperties** utility might display a message from the application server (WASX7357I). You can ignore this message.
3. When you supply a password in a text file for **changeSecurityProperties**, there is a small security exposure. When you enter a password in the file, the password is entered in clear (unencrypted). After you have run **changeSecurityProperties**, the password remains in clear in the text file you have edited, but if you run **showSecurityProperties** the password is output encrypted. Thus, your potential security exposure is limited to the time from when you entered the password in the text file until when you manually deleted the text file after using **changeSecurityProperties**.

   Attention: If you subsequently want to change *other* parameters and do *not* want to change any passwords, you must perform one of the following actions before running **changeSecurityProperties**. This is to prevent that the row of asterisks is applied as the password:

   - Resupply the passwords in clear.
   - Comment the password properties.
   - Delete the password properties.

Action 6 - update SOAP properties

About this task

After the password of the WebSphere Application Server administration user has been modified, it is important to change the SOAP client properties using the **updateWas.sh/.bat** script (see "Application server - updating the SOAP properties after changing the WebSphere Application Server user or its password" on page 453 for full details). For example:

```
updateWas.sh -user john.smith@domain.com -password zzzz
```

where the **user** and **password** options are optional and represent the user that is authorized to stop the WebSphere Application Server.

Alternatively, you can supply a credentials file containing the new credentials. The contents of the file specified are parsed to locate the credentials and then the `soap.client.props` file is updated with the new credentials. The following is the usage for the **updateWas.sh** script:

```
updateWas(-credentialfile filepath | -user username -password password)
```

The format of the credentials file is as follows:

```
user=SampleUser
password=SamplePassword
wasuser=WasUser
wasPassword=WasPassword
error=Error
```

Stop and restart WebSphere Application Server using the **stopappserver** and **startappserver** commands to make the change effective.

Action 7 - Windows - update Windows services

About this task

On Windows, the <*TWS_user*> account is used to start the following services:

- IBM Token Service for <*TWS_user*>
- IBM Workload Scheduler for <*TWS_user*>
- IBM WebSphere Application Server V7.0 - <*TWS_user*>.

The password must be updated in the properties of these services, or they are not able to start at next reboot. This is done as follows:

1. Stop all IBM Workload Scheduler processes. See "Unlinking and stopping IBM Workload Scheduler" on page 430 for details.
2. Locate the script **updateWasService.bat** in the `>TWA_home>/wastools` directory.
3. Run **updateWasService.bat**, as described in "Application server - updating the Windows services after modifications" on page 452, giving the new password as the <*WAS_user_password*>.
4. Restart all IBM Workload Scheduler processes using the **StartUp** command.

Action 8 - change the IBM Workload Scheduler user definition

About this task

If the user ID is used within IBM Workload Scheduler to run jobs, follow this procedure:

1. Run the **composer modify user** command. The user details of the selected user are written to a temporary file, which is opened.
2. Edit the password field so that it contains the new password value delimited by double quotation marks characters (").
3. Save the file, and the contents are added to the database.
4. To make the change immediately effective in the current plan, issue the **conman altpass** command.

For the full syntax of these commands see the *IBM Workload Scheduler: User's Guide and Reference*.

Using the changePassword script

About this task

Use the **changePassword** script from the <*TWA_home*>`/wastools` directory to change the passwords of any of the following users:

- IBM Workload Scheduler instance owner (<*TWS_user*>)
- WebSphere Application Server user
- Database (J2C) user for either Oracle or DB2
- Streamlogon user (Windows only)

Before changing any passwords, you must first change the password at the operating system level using native commands, as follows:

On UNIX operating systems
use the **passwd** command.

On Windows operating systems
use the **net user** command.

If you use special characters in the password, ensure you use a "\" (backslash) before the special character. The following rules apply:

On Windows operating systems:
Passwords for users can include any alphanumeric characters and ()!?=^*/~[]$_+;:.,@`-#.

On UNIX and LINUX systems:
Passwords for users can include any alphanumeric characters and ()!?=*~_+.-.

If required, the script performs the necessary changes to the *useropts* file and stops and restarts the WebSphere Application Server. You can run this script from your master domain manager or IBM Workload Scheduler agent. The script determines the role of the users for which the password must be changed and performs the checks and actions of the manual procedure described in actions 1 through 8. Run the script as follows:

UNIX

```
changePassword.sh  (-credentialfile <FILEPATH> |
                     -user <USERID>
                     -password <PASSWORD>
                    [-wasuser <WASUSER>]
                    [-waspassword <WASPASSWORD>])
                    [-usroptshome <HOMEDIR>]
```

Where the arguments are as follows:

-credentialfile <*FILEPATH*
This argument is optional. You can provide the new credentials in a file represented by <*filepath*>. The script parses the file and detects the new credentials for the IBM Workload Scheduler administrator user, and the WebSphere Application Server user. It then updates the `soap.client.props` file located in the properties directory of the WebSphere Application Server profile with the new credentials. The format for the credentials file is as follows:

```
user=SampleUser
password=SamplePassword
wasuser=WasUser
wasPassword=WasPassword
error=Error
```

–user <*USERID*>
This argument is optional. Specify the user whose password you are changing.

–password <*PASSWORD*>
This argument is optional. Specify the new password for the user.

–wasuser <*WASUSER*>
This argument is optional. Specify the WebSphere Application Server user. By default, the <*USERID*> value is used.

–waspassword <*WASPASSWORD*>
This argument is optional. Specify the WebSphere Application Server user password. By default, the <*PASSWORD*> value is used. The <*WASUSER*> and <*WASPASSWORD*> values are ignored if the WebSphere Application Server is not present on this instance or if the script is running for a IBM Workload Scheduler instance.

–usroptshome<*HOMEDIR*>
This argument is optional. The script searches for the `USEROPTS` file

in the *TWA_home*/*.TWS*WebSphere Application Server directory. This argument is ignored if the script is not running for a IBM Workload Scheduler instance.

Windows

```
changePassword.bat  (-credentialfile <FILEPATH> |
                     -user <USERID>
                     -password <PASSWORD>
                     [-srvuser <SRVUSERID>]
                     [-srvpassword <SRVPASSWORD>]
                     [-wasuser <WASUSERID>]
                     [-waspassword <WASPASSWORD>])
                     [-usroptshome <HOMEDIR>]
                     [-streamlogonws <WS>]
                     [-streamlogondm <DOMAIN>]
```

Where the arguments are as follows:

-credentialfile <*FILEPATH*>
This argument is optional. You can provide the new credentials in a file represented by <*filepath*>. The script parses the file and detects the new credentials for the IBM Workload Scheduler administrator user, the Windows service user, and the WebSphere Application Server user. It then updates the `soap.client.props` file located in the properties directory of the WebSphere Application Server profile with the new credentials. The format for the credentials file is as follows:

```
user=SampleUser
password=SamplePassword
wasuser=WasUser
wasPassword=WasPassword
error=Error
```

A sample file named, `cred_TEMPLATE.properties`, can be found in the `wastools` folder.

–user <*USERID*>
This argument is optional. Specify the user whose password you are changing.

–password <*PASSWORD*>
This argument is optional. Specify the new password for the user.

–srvuser <*SRVUSERID*>
This argument is optional. Specify the Windows service user. If you are running the script for a IBM Workload Scheduler instance, the value specified here is the same as the IBM Workload Scheduler user. By default, the <*USERID*> value is used.

–srvpassword <*SRVPASSWORD*>
This argument is optional. The password of the Windows service user. By default, the <*PASSWORD*> value is used.

–wasuser <*WASUSER*>
This argument is optional. Specify the WebSphere Application Server user. By default, the <*USERID*> value is used.

–waspassword <*WASPASSWORD*>
This argument is optional. Specify the WebSphere Application Server user password. By default, the <*PASSWORD*> value is used. The <*WASUSER*> and <*WASPASSWORD*> values are ignored if the WebSphere Application Server is not present on this instance or if the script is running for a IBM Workload Scheduler instance.

–usroptshome<*HOMEDIR*>
This argument is optional. The script searches for the USEROPTS file in the *TWA_home*/*.TWS*WebSphere Application Server directory. By default, the home directory of the user running the script is used.

–streamlogonws<*WS*>
This argument is optional (Windows only). The script updates the user definition for the <WS>#<DOMAIN>/<USER> user in the IBM Workload Scheduler database. By default, the user definition for the <USER> is updated. The update is performed only if the tool is running on the master domain manager in a Windows environment.

–streamlogondm<*DOMAIN*>
This argument is optional. Specify the domain of the user specified for the <USER>.

Unlinking and stopping IBM Workload Scheduler

About this task

Before you perform an upgrade or uninstall, install a fix pack, or perform maintenance activities, ensure that all IBM Workload Scheduler processes and services are stopped. Follow these steps:

1. If you have jobs that are currently running on the workstation, wait for them to finish. To determine which are not finished, check for jobs that are in the *exec* state. When there are no jobs in this state, and you have allowed sufficient time for all events to be distributed in your network, you can continue with the rest of the procedure.
2. If the workstation that you want to stop is not the master domain manager, unlink the workstation by issuing the following command from the command line of the master domain manager:
   ```
   conman "unlink workstationname;noask"
   ```
3. Stop the WebSphere Application Server by using the conman stopappserver command (see "Starting and stopping the application server and **appservman**" on page 450)
4. All IBM Workload Scheduler processes on the workstation must then be stopped manually. From the command line, while logged on as the <*TWS_user*>, enter the following command:
   ```
   conman "stop;wait"
   ```
5. From the command line, stop the netman process as follows:

   UNIX Run the conman "shut" command.

   Note: Do not use the UNIX kill command to stop IBM Workload Scheduler processes.

   Windows
   Fom the IBM Workload Scheduler home directory, run the shutdown.cmd command.
6. If the workstation is at version 8.4 or later, stop the SSM agent, as follows:
   - On Windows, stop the service IBM Workload Scheduler SSM Agent (for <*TWS_user*>).
   - On UNIX, run the stopmon command.
7. If you are updating an agent, remove (unmount) any NFS mounted directories from the master domain manager.

8. If you are upgrading an installation that includes the connector, stop the connector.

To verify if there are services and processes still running:

UNIX Enter the command:

```
ps -u <TWS_user>
```

Verify that the following processes are not running: netman, mailman, batchman, writer, jobman, JOBMAN, stageman, logman, planman, monman, ssmagent.bin, and appservman.

Windows

Run **Task Manager**, and verify that the following processes are not running: netman, mailman, batchman, writer, jobman, stageman, JOBMON, tokensrv, batchup, logman, planman, monman, ssmagent, and appservman.

Also, ensure that no system programs are accessing the directory or its subdirectories, including the command prompt and Windows Explorer.

Changing the database host name, port, or database name

To change the database host name, port, or database name, the procedure differs, depending on whether the database is on DB2 or Oracle:

- "Change the DB2 host name, port, or database name"
- "Changing the Oracle host name, port, or database name" on page 437

Change the DB2 host name, port, or database name

About this task

If you need to change the DB2 host name, port, or database name, use the **changeDataSourceProperties** utility to reflect these changes in the application server on the master domain manager.

When you installed IBM Workload Scheduler, the default database name that was used for the creation of the database was *TWS* (which you might have changed). You also supplied the port and the host name of the DB2 server. If you want to change any of these details, you must do the following:

1. Stop DB2 and IBM Workload Scheduler
2. Use the facilities of DB2 (see DB2 documentation for details) or the operating system to change the database name, port or host name.
3. Change the configuration of the IBM Workload Scheduler application server so that it points correctly to the changed DB2 configuration.

   The procedure requires you to stop the application server, create a text file of the current data source properties, edit the file, run the utility and restart the application server. Find information about how to do this as follows:

   - "Application server - using the utilities that change the properties" on page 459 gives a generic description of the procedure for making any change to the WebSphere Application Server properties
   - "Changing data source properties" on page 432 lists all the data source properties and gives other reference information about the utility
   - When editing the text file of the current data source properties:
     a. Edit the text file and locate the following properties:

```
############################################################
# DB2 Type4 Resource Properties
############################################################
DB2Type4DatabaseName=TWS
DB2Type4ServerName=localhost
DB2Type4PortNumber=50083
```

b. Set the following entries:

DB2Type4DatabaseName
The new name of the IBM Workload Scheduler database.

DB2Type4ServerName
The new DB2 server host name.

DB2Type4PortNumber
The new DB2 server port.

When you change a DB2 server port, you must also modify the configuration of the node where the IBM Workload Scheduler was cataloged:

– If you are working with a DB2 client, open a command line session and log in as DB2 Administrator, then run the following commands:

```
DB2 CLIENT
db2 uncatalog node <TWSDBNAME>_ND
db2 catalog tcpip node <TWSDBNAME>_ND remote <HOSTNAME>
server <NEWPORT>
```

– If you are working with a DB2 server, open a command line session and log in as DB2 Administrator, then run the following commands :

```
DB2 SERVER
db2 uncatalog node LBNODE
db2 catalog tcpip node LBNODE remote 127.0.0.1 server <NEWPORT>
```

Do not change any other properties.

Note: The utility might display a message from the application server (WASX7357I:). You can ignore this message.

4. Start DB2 and IBM Workload Scheduler.

This script can also be used to change other data source properties. However, if you do so, IBM Workload Scheduler might not work correctly. You are advised to make any other changes only under the instructions of IBM Software Support, to correct specific problems. One of those specific problems could be the need to resolve problems with the JBDC driver, see “Resolving problems with the JDBC driver” on page 436.

Changing data source properties

You run the **changeDataSourceProperties** script on the master domain manager to change the data source properties of the RDBMS in use with the master domain manager. You are required to update the data source properties in the following cases:

- You migrate your data from an Oracle database to DB2 using the reconfiguration method.
- You change the database name, server, host, or port.
- You change the path to the JDBC driver of the RDBMS.

The procedure for running the script is described in detail in “Application server - using the utilities that change the properties” on page 459, but in summary you do the following:

- Run **showDataSourceProperties.sh (.bat) > my_file_name** to obtain the current properties
- Edit *my_file_name*
- Run **changeDataSourceProperties.sh (.bat) my_file_name**

Note: *my_file_name* must be the fully qualified path of the file.

The change utility calls the **wsadmin** utility by running **ChangeDataSourceProperties.jacl** with the properties file as input.

Only the WebSphere Application Server `resources.xml` are affected by this script. The full path of the file is:

```
<WAS_profile_path>/config/cells/TWSNodeCell/nodes/
  TWSNode/servers/server1/resources.xml
```

where the default value for <*WAS_profile_path*> is <TWA_home>/WAS/TWSprofile.

The following example shows the properties that can be changed with this utility. Only some of the options listed are currently in use by IBM Workload Scheduler.

```
############################################################
JDBC Path Variables
############################################################
ORACLE_JDBC_DRIVER_PATH=
DB2_JDBC_DRIVER_PATH=c:/ibm/sqllib/java
DB2UNIVERSAL_JDBC_DRIVER_PATH=c:/ibm/sqllib/java


############################################################
DB2 Type2 Resource Properties
############################################################
DB2Type2JndiName=
DB2Type2Description=
DB2Type2ConnectionAttribute=cursorhold=0
DB2Type2EnableMultithreadedAccessDetection=false
DB2Type2Reauthentication=false
DB2Type2JmsOnePhaseOptimization=false
DB2Type2DatabaseName=TWSZ_DB
DB2Type2PreTestSQLString=SELECT 1 FROM SYSIBM.SYSDUMMY1


############################################################
DB2 Type4 Resource Properties
############################################################
DB2Type4JndiName=jdbc/twsdb
DB2Type4DatabaseName=TWSZ
DB2Type4DriverType=4
DB2Type4ServerName=myhost.mydomain.com
DB2Type4PortNumber=50000
DB2Type4SslConnection=false
DB2Type4Description=
DB2Type4TraceLevel=
DB2Type4TraceFile=
DB2Type4FullyMaterializeLobData=true
DB2Type4ResultSetHoldability=2
DB2Type4CurrentPackageSet=
DB2Type4ReadOnly=false
DB2Type4DeferPrepares=true
DB2Type4CurrentSchema=
DB2Type4CliSchema=
DB2Type4RetrieveMessagesFromServerOnGetMessage=true
DB2Type4ClientAccountingInformation=
```

```
DB2Type4ClientApplicationInformation=
DB2Type4ClientUser=
DB2Type4ClientWorkstation=
DB2Type4CurrentPackagePath=
DB2Type4CurrentSQLID=
DB2Type4KerberosServerPrincipal=
DB2Type4LoginTimeout=0
DB2Type4SecurityMechanism=
DB2Type4TraceFileAppend=false
DB2Type4CurrentFunctionPath=
DB2Type4CursorSensitivity=
DB2Type4KeepDynamic=
DB2Type4CurrentLockTimeout=
DB2Type4EnableMultithreadedAccessDetection=false
DB2Type4Reauthentication=false
DB2Type4JmsOnePhaseOptimization=false
DB2Type4PreTestSQLString=SELECT 1 FROM SYSIBM.SYSDUMMY1
DB2Type4DbFailOverEnabled=false
DB2Type4ConnRetriesDuringDBFailover=100
DB2Type4ConnRetryIntervalDuringDBFailover=3000
# DB2Type4IsolationLevel can be one of the following:
#       CURSOR_STABILITY or READ_STABILITY
DB2Type4IsolationLevel=CURSOR_STABILITY


################################################################
Oracle Type2 Resource Properties
################################################################
OracleType2JndiName=
OracleType2DriverType=
OracleType2URL=jdbc:oracle:oci:@ORCL
OracleType2DatabaseName=
OracleType2ServerName=
OracleType2PortNumber=1521
OracleType2OracleLogFileSizeLimit=0
OracleType2OracleLogFileCount=1
OracleType2OracleLogFileName=
OracleType2OracleLogTraceLevel=INFO
OracleType2OracleLogFormat=SimpleFormat
OracleType2OracleLogPackageName=oracle.jdbc.driver
OracleType2TNSEntryName=
OracleType2NetworkProtocol=
OracleType2DataSourceName=
OracleType2LoginTimeout=
OracleType2Description=
OracleType2EnableMultithreadedAccessDetection=false
OracleType2Reauthentication=false
OracleType2JmsOnePhaseOptimization=false
OracleType2PreTestSQLString=SELECT 1 FROM DUAL
OracleType2DbFailOverEnabled=false
OracleType2ConnRetriesDuringDBFailover=100
OracleType2ConnRetryIntervalDuringDBFailover=3000


################################################################
Oracle Type4 Resource Properties
################################################################
OracleType4JndiName=
OracleType4DriverType=
OracleType4URL=jdbc:oracle:thin:@//localhost:1521/ORCL
OracleType4DatabaseName=
OracleType4ServerName=
OracleType4PortNumber=1521
OracleType4OracleLogFileSizeLimit=0
OracleType4OracleLogFileCount=1
OracleType4OracleLogFileName=
OracleType4OracleLogTraceLevel=INFO
```

```
OracleType4OracleLogFormat=SimpleFormat
OracleType4OracleLogPackageName=oracle.jdbc.driver
OracleType4TNSEntryName=
OracleType4NetworkProtocol=
OracleType4DataSourceName=
OracleType4LoginTimeout=
OracleType4Description=
OracleType4EnableMultithreadedAccessDetection=false
OracleType4Reauthentication=false
OracleType4JmsOnePhaseOptimization=false
OracleType4PreTestSQLString=SELECT 1 FROM DUAL
OracleType4DbFailOverEnabled=false
OracleType4ConnRetriesDuringDBFailover=100
OracleType4ConnRetryIntervalDuringDBFailover=3000
```

When you change data source properties, observe the following rules:

- If a property is not provided in the properties file, the current value is not changed
- If a property is provided with a non-blank value, the current value is updated.
- If a property is provided with a blank value, the setting is set to blank if the property is classified as erasable or left unchanged if not.
- Always use type 4 data sources for DB2 and type 2 data sources for Oracle.
- Set the appropriate JDBC driver path variable for the RDBMS of your choice.
  - For DB2, the JDBC driver is located in the `java` subfolder of the `sqllib` directory. For example:

    ```
    DB2_JDBC_DRIVER_PATH=c:/program files/ibm/sqllib/java
    ```

    or

    ```
    DB2UNIVERSAL_JDBC_DRIVER_PATH=c:/program files/ibm/sqllib/java
    ```
  - For Oracle, it is located in the `jdbc/lib` subfolder of the Oracle home directory. For example:

    ```
    ORACLE_JDBC_DRIVER_PATH=C:/Oracle/product/10.2.0/db_1/jdbc/lib
    ```
- Make sure that the data source JNDI name is always set to `jdbc/twsdb` in the `...JndiName` property of the RDBMS you use. If you change the RDBMS, proceed as follows:
  1. Reset to a name of your choice the `...JndiName` property of the RDBMS from which you are changing.
  2. Set to `jdbc/twsdb` the `...JndiName` property of the new RDBMS.
- See that the following properties are set:
  - For DB2:

    ```
    DB2Type4JndiName
    DB2Type4DatabaseName
    DB2Type4ServerName
    DB2Type4PortNumber
    ```
  - For Oracle:

    ```
    OracleType2JndiName
    OracleType2DatabaseName
    OracleType2ServerName
    OracleType2PortNumber
    ```

Displaying the current data source properties: To display the current properties, use the following utility:

UNIX **`showDataSourceProperties.sh`**

Windows
`showDataSourceProperties.bat`

Resolving problems with the JDBC driver

IBM Workload Scheduler is supplied using the JDBC driver type 4 for DB2 and type 2 for Oracle. However, each can use the other driver type, if necessary. IBM Software Support might ask you to change to this driver. This section tells you how.

Attention: This procedure must only be performed under the control of IBM Software Support.

To change the driver you need to change the data source properties following the procedure described in "Changing the database host name, port, or database name" on page 431. However, the parameters that you change are different. This is an example of the type 4 and type 2 parameters for DB2:

JDBC driver type 4 parameters

```
################################################################
# DB2 Type4 Resource Properties
################################################################
DB2Type4JndiName=jdbc/twsdb
DB2Type4DatabaseName=TWSZ
DB2Type4DriverType=4
DB2Type4ServerName=myhost.mydomain.com
DB2Type4PortNumber=50000
DB2Type4SslConnection=false
DB2Type4Description=
DB2Type4TraceLevel=
DB2Type4TraceFile=
DB2Type4FullyMaterializeLobData=true
DB2Type4ResultSetHoldability=2
DB2Type4CurrentPackageSet=
DB2Type4ReadOnly=false
DB2Type4DeferPrepares=true
DB2Type4CurrentSchema=
DB2Type4CliSchema=
DB2Type4RetrieveMessagesFromServerOnGetMessage=true
DB2Type4ClientAccountingInformation=
DB2Type4ClientApplicationInformation=
DB2Type4ClientUser=
DB2Type4ClientWorkstation=
DB2Type4CurrentPackagePath=
DB2Type4CurrentSQLID=
DB2Type4KerberosServerPrincipal=
DB2Type4LoginTimeout=0
DB2Type4SecurityMechanism=
DB2Type4TraceFileAppend=false
DB2Type4CurrentFunctionPath=
DB2Type4CursorSensitivity=
DB2Type4KeepDynamic=
DB2Type4CurrentLockTimeout=
DB2Type4EnableMultithreadedAccessDetection=false
DB2Type4Reauthentication=false
DB2Type4JmsOnePhaseOptimization=false
DB2Type4PreTestSQLString=SELECT 1 FROM SYSIBM.SYSDUMMY1
DB2Type4DbFailOverEnabled=false
DB2Type4ConnRetriesDuringDBFailover=100
DB2Type4ConnRetryIntervalDuringDBFailover=3000
# DB2Type4IsolationLevel can be one of the following:
#        CURSOR_STABILITY or READ_STABILITY
DB2Type4IsolationLevel=CURSOR_STABILITY
```

JDBC Driver type 2 parameters

```
################################################################
# DB2 Type2 Resource Properties
################################################################
```

```
DB2Type2JndiName=
DB2Type2Description=
DB2Type2ConnectionAttribute=cursorhold=0
DB2Type2EnableMultithreadedAccessDetection=false
DB2Type2Reauthentication=false
DB2Type2JmsOnePhaseOptimization=false
DB2Type2DatabaseName=TWSZ_DB
DB2Type2PreTestSQLString=SELECT 1 FROM SYSIBM.SYSDUMMY1
```

Switching drivers or changing the JNDI name: The data source JNDI name must be unique. In the above examples the JNDI name for driver 4 is set to the correct value. To switch drivers, modify the parameters so that the values are reversed, as follows:

```
Example 1: default values for the JNDI name:

#DB2Type4JndiName=jdbc/twsdb

...

#DB2Type2JndiName=jdbc/twsdb2



Example 2: switched values for the JNDI name:

#DB2Type4JndiName=jdbc/twsdb2

...

#DB2Type2JndiName=jdbc/twsdb
```

To change the driver names to a different value, see the following:

```
Example 3: different values for the JNDI name:

#DB2Type4JndiName=jdbc/twsdb_test4

...

#DB2Type2JndiName=jdbc/twsdb_test2
```

Changing the Oracle host name, port, or database name

About this task

If you need to change the Oracle host name, port, or database name, you can normally manage the change within Oracle. This is because WebSphere Application Server points at the Oracle service where these items are defined. See the Oracle documentation for information on how to change them.

However, the properties that you are changing might be defined in *<WAS_profile_path>*/properties/TWSConfig.properties, where *WAS_profile_path* corresponds to the WebSphere Application Server profile path you specified at installation time. The default path is *TWA_home*/WAS/TWSprofile. In this case you must ensure that they are changed here, as well. The properties in question are:

```
com.ibm.tws.dao.rdbms.rdbmsName = ORACLE
com.ibm.tws.dao.rdbms.modelSchema = <TWS_Oracle_User>
com.ibm.tws.dao.rdbms.eventRuleSchema = <TWS_Oracle_User>
com.ibm.tws.dao.rdbms.logSchema = <TWS_Oracle_User>
```

Changing the workstation host name or IP address

When you change the host name, the IP address or both on the workstations of your IBM Workload Scheduler environment to have it function properly, you must report the changed value on:

- The WebSphere Application Server if the following components changed the host name, the IP address or both:
  - Master domain manager
  - Backup master domain manager
  - Connector or z/OS connector
  - Dynamic Workload Console

  For more information see "Reporting the changes in the WebSphere Application Server configuration file."
- The following components if the workstation where you installed the RDBMS changed the host name, the IP address or both:
  - Master domain manager
  - Backup master domain manager
  - Dynamic domain manager
  - Backup dynamic domain manager

  For more information see "Reporting the changed host name or IP address of the workstation where you installed the RDBMS" on page 440.
- The workstation definitions if you installed the following components:
  - Master domain manager
  - Backup master domain manager
  - Dynamic domain manager
  - Backup dynamic domain manager
  - Fault-tolerant agent and standard agent
  - Domain manager

For more information see "Reporting the changed host name or IP address in the workstation definition" on page 441.

Reporting the changes in the WebSphere Application Server configuration file

About this task

If the following components changed the host name or IP address you must report the changed value in the WebSphere Application Server configuration file, performing the following steps:

- Master domain manager
- Backup master domain manager
- Connector or z/OS connector
- Dynamic Workload Console

1. Stop the WebSphere Application Server.
2. Obtain the changed host name, IP address, or both.
3. Run the **showHostProperties** tool by redirecting the output to a text file to obtain the current properties.
4. Open the file and go to the Host Configuration Panel.

   Below an example of the section you have to refer to:

```
############################################################
# Host Configuration Panel
############################################################
# Old Hostname
```

```
oldHostname=myoldhost.romelab.ibm.it.com
# New Hostname
newHostname=mynewhost.romelab.ibm.it.com.....

############################################################
Ports Configuration Panel
############################################################
bootPort=41117
bootHost=myhost.mydomain.com
soapPort=41118
soapHost=myhost.mydomain.com
httpPort=41115
httpHost=*
httpsPort=41116
httpsHost=*
adminPort=41123
adminHost=*
adminSecurePort=41124
adminSecureHost=*
sasPort=41119
sasHost=myhost.mydomain.com
csiServerAuthPort=41120
csiServerAuthHost=myhost.mydomain.com
csiMuthualAuthPort=41121
csiMuthualAuthHost=myhost.mydomain.com
orbPort=41122
orbHost=myhost.mydomain.com
```

5. Verify that the value for the properties listed below were changed with the actual values:
   - Old Hostname
   - New Hostname
   - The host names for the specific port properties

   If this values are different from the actual host name or IP address values, proceed with Step 6. If this values are not changed skip the steps below.
6. Modify the values of the properties by running the **changeHostProperties** tool. For more information, see "Application server - changing the host name or TCP/IP ports" on page 455.
7. Restart the WebSphere Application Server if you do not need to perform the RDBMS changes. To perform the RDBMS changes, see "Reporting the changed host name or IP address of the workstation where you installed the RDBMS" on page 440.
8. Propagate the changes to the interfaces as follows:

   Address of the master domain manager changes
   - On each fault-tolerant agent, dynamic agent, and standard agent you configured to connect to the **comman** command line, update the **host** parameter present in the "Attributes for CLI connections" section in the `localopts` file. Usually you have the **host** parameter defined in the `localopts` file of the workstations you use to submit predefined jobs and job streams (sbj and sbs commands).
   - On every command line client, update the **host** parameter present in the "Attributes for CLI connections" section in the `localopts` file.
   - On the Dynamic Workload Console update the engine connections.

   Address of the Dynamic Workload Console changes
   Notify all the users of the new web address.

Reporting the changed host name or IP address of the workstation where you installed the RDBMS

About this task

If you changed the host name or IP address in the workstation where you installed the RDBMS, contact your database administrator to reconfigure your RDBMS to use the new host name or IP address. If you are using DB2, see the procedure described in https://www-304.ibm.com/support/docview.wss?uid=swg21258834.

Propagate the changes to the following components by performing the steps below:

- Master domain manager
- Backup master domain manager
- Dynamic domain manager
- Backup dynamic domain manager

1. Stop the WebSphere Application Server.
2. Run the **showDataSourceProperties** tool by redirecting the output to a text file to obtain the current properties.
3. Open the file and identify the database section where the *databasetype*Type*n*JndiName property is equal to **jdbc/twsdb**.

   where *databasetype* is the database you are using, for example DB2, and *n* can be 2 or 4.

   Below an example of the section you have to refer to if you are using DB2:

   ```
   ############################################################
   DB2 Type4 Resource Properties
   ############################################################
   DB2Type4JndiName=jdbc/twsdb
   DB2Type4DatabaseName=TWSZ
   DB2Type4DriverType=4
   DB2Type4ServerName=myhost.mydomain.com
   .....
   ```
4. Verify the value of the *databasetype*Type*n*ServerName property. If this value is changed proceed with Step 5. If this value is not changed skip the steps below.
5. Modify the *databasetype*Type*n*ServerName=*value* property by running **changeDataSourceProperties**. For more information, see "Changing the database host name, port, or database name" on page 431.
6. To use the:

   Dynamic Workload Console reports
   Update the database connections.

   Command line reports
   Update the following section of the <*report_home*>\config\common.properties file

   ```
   ######################################################################
   # DATABASE PROPERTIES
   ######################################################################
   # Specify the host name or TCP/IP address of the database,
   # its port number and name.
   DatabaseHostname=<hostname>
   DatabasePort=50000
   DatabaseName=TWS
   ........................
   ```

   Where <*report_home*> is the directory where you extract the package.
7. Restart the WebSphere Application Server.

Reporting the changed host name or IP address in the workstation definition

About this task

Run this procedure if you changed the host name or IP address on the following components:
- Master domain manager
- Backup master domain manager
- Fault-tolerant agent and standard agent
- Domain manager

To modify the host name or the IP address on the workstation definition, perform the following steps:

1. Use **composer** or the Dynamic Workload Console to check the workstation definition stored in the database for the IBM Workload Scheduler instance installed on the workstation where the IP address or the host name changed.
2. Verify the **node** attribute contains the new host name or IP address. If this value is changed proceed with Step 3. If this value is not changed skip the steps below.
3. Change the value of the **node** parameter with the new value.
4. Refresh the new workstation definition into the plan. Do it immediately if you are changing the host name or the IP address of a master domain manager or a domain manager. If you are changing them on a workstation that is not a master domain manager or a domain manager you can wait the next scheduled plan generation to refresh your workstation definition in the Symphony file. In this case during this production day you cannot run jobs on this workstation. To generate the plan, perform the following steps:
   a. Run **optman ls** and take note of the actual value of the **enCarryForward** parameter.
   b. If this value is not set to **all**, run
      ```
      optman chg cf=ALL
      ```
      to set it to **all**
   c. Add the new workstation definition to the plan, by running:
      ```
      JnextPlan –for 0000
      ```
   d. Reassign the original value to the **enCarryForward** parameter.

Reporting the changed host name or IP address of the dynamic workload broker server

About this task

The dynamic workload broker server is a component that IBM Workload Scheduler installs when you install the following components:
- Master domain manager
- Backup master domain manager
- Dynamic domain manager
- Backup dynamic domain manager

If you changed the host name or the IP address on the dynamic workload broker server, or if you installed a new one run the procedure described in “Reporting the changes in the WebSphere Application Server configuration file” on page 438.

If you changed the host name or the IP address on a master domain manager or backup master domain manager and you ran the "Reporting the changes in the WebSphere Application Server configuration file" on page 438 procedure, skip this section.

If you changed the host name or the IP address on the dynamic domain manager or backup dynamic domain manager you do not need to change the definition of your broker workstation (type **broker**), because the value of the **node** attribute is set to the *localhost* value to allow to switch between the dynamic workload broker server and its backup.

After you ran the procedure, propagate the changes to the dynamic agent and update the **ResourceAdvisorURL** property in the `JobManager.ini` file on each agent connected to that dynamic workload broker server, by performing the following steps:

1. Run the following command to stop the agent:
   ```
   ShutDownLwa
   ```
2. Edit the `JobManager.ini` file and change the host name or the IP address in the **ResourceAdvisorURL** property.
3. Run the following command to start the agent:
   ```
   StartUpLwa
   ```

Perform the following changes:

1. Open the `JobDispatcherConfig.properties` file and change the value of the **JDURL=https://*host_name*** property to reflect the new host name or IP address.
2. Open the `CliConfig.properties` file and change the value of the **ITDWBServerHost=/*host_name*** property to reflect the new host name or IP address.
3. Open the `ResourceAdvisorConfig.properties` file and change the value of the **ResourceAdvisorURL=https://*host_name*** property to reflect the new host name or IP address.
4. From the *<TWA_home>*`/TDWB/bin` directory, run the following command:

   On Windows operating systems:
   ```
   exportserverdata.bat
   ```

   On UNIX and Linux operating systems:
   ```
   exportserverdata.sh
   ```

   This command extracts a list of URIs (Uniform Resource Identifier) of all the dynamic workload broker instances from the IBM Workload Scheduler database and copies them to a temporary file. By default, the list of URIs is saved to the `server.properties` file, located in the current directory.
5. Change all the entries that contain the old host name to reflect the new host name.
6. Place the file back in the database, by running the following command:

   On Windows operating systems:
   ```
   importserverdata.bat
   ```

   On UNIX and Linux operating systems:
   ```
   importserverdata.sh
   ```
7. Stop the dynamic workload broker, by running the following command:
   ```
   StopBrokerApplication
   ```
8. Start the dynamic workload broker, by running the following command:

```
StartBrokerApplication
```

Reporting the changed host name or IP address of the dynamic agent

About this task

If you changed the host name or the IP address on the workstation where you installed the dynamic agent, the changes are automatically reported stopping and starting the agent using the following commands:

1. To stop the agent:
   ```
   ShutDownLwa
   ```
2. To start the agent:
   ```
   StartUpLwa
   ```

Note: Do not modify manually the value of the **node** parameter in the dynamic agent workstation definition.

Changing the security settings

This section describes how to modify the security settings of IBM Workload Scheduler.

About this task

Use the **changeSecurityProperties** script located in *TWA_home*/TWS/wastool to change various security settings on the application server. For the settings related to SSL, see Chapter 7, "Setting connection security," on page 303. For the settings related to the passwords of the database access users, see "Changing key IBM Workload Scheduler passwords" on page 419. You can also change other settings, such as the active user registry or the local operating system ID and password.

The procedure requires you to stop the application server, create a text file of the current security properties, edit the file, run the utility, and restart the application server.

- For more information about the procedure to make any changes to the WebSphere Application Server properties, see "Application server - using the utilities that change the properties" on page 459.
- To determine which properties are to be changed, see:
  - Chapter 5, "Configuring authentication," on page 245, for information about the properties to be changed to modify your user registry configuration for user authentication.
  - "Scenario: Connection between the Dynamic Workload Console and the IBM Workload Scheduler component that has a distributed connector" on page 304, for information about the properties to be changed to configure SSL communication between the different interfaces and the IBM Workload Scheduler engine.
  - "Migrating data from DB2 to Oracle and *vice versa*" on page 372, for information about the properties to be changed when migrating your database from one database platform to another.
  - "Changing key IBM Workload Scheduler passwords" on page 419, for information about how to use the properties to determine the procedure required for changing key passwords.

- To change the text file of the current security properties, perform the following steps:
  1. Edit the text file and locate the properties you need to change.
  2. Make any required changes to the properties.

     Do not change any other properties.

  Note:

  1. The utility might display a message from the application server (WASX7357I:). You can ignore this message.
  2. When you supply a password in a text file for **changeSecurityProperties**, there is a small security exposure. When you enter a password in the file, the password is entered in clear (unencrypted). After you have run **changeSecurityProperties**, the password remains in clear in the text file you have edited, but if you run **showSecurityProperties** the password is output encrypted, as a row of asterisks. Thus, your potential security exposure is limited to the time from when you entered the password in the text file until when you manually deleted the text file after using **changeSecurityProperties**.

     Attention: If you want to change parameters *other* than a password, and do *not* want to change a password, ensure that you perform one of the following actions before running **changeSecurityProperties**. This is required to prevent that the row of asterisks is applied as the password:

     – Resupply the passwords in clear.
     – Comment the password properties.
     – Delete the password properties.

Managing the event processor

About this task

The only maintenance issue for the event processor is the management of the EIF event queue, `cache.dat`. The event queue is circular, with events being added at the end and removed from the beginning. However, if there is no room to write an event at the end of the queue it is written at the beginning, overwriting the event at the beginning of the queue.

To increase the size of the event processor queue, follow this procedure:

1. At the workstation running the event processor, locate the file:

   `<WAS_profile_path>/temp/TWS/EIFListener/eif.templ`

   where *WAS_profile_path* corresponds to the WebSphere Application Server profile path you specified at installation time. The default path is *TWA_home*`/WAS/TWSprofile`.
2. Edit the file and locate the keyword:

   `BufEvtMaxSize`
3. Increase the value of this keyword, according to your requirements.
4. Stop and restart the WebSphere Application Server using the **conman stopappserver** and **conman startappserver** commands (see "Starting and stopping the application server and **appservman**" on page 450).

Starting, stopping, and displaying dynamic workload broker status

About this task

To start or stop dynamic workload broker, use the **startBrokerApplication** or **stopBrokerApplication** commands on the active master domain manager. Since these commands are processed asynchronously, the **brokerApplicationStatus** command allows you to check the status of dynamic workload broker following a start or a stop. Ensure that WebSphere Application Server is running and proceed as follows:

Starting dynamic workload broker

Use **startBrokerApplication.sh** on UNIX and Linux or **startBrokerApplication.bat** on Windows as follows:

startBrokerApplication -user *username* -password *password* [-port *portnumber*]

where *username* and *password* are the credentials used during the installation. The parameter *portnumber* is optional; this value is defined in the `Broker.Workstation.Port` property in `/opt/IBM/TWA92_SVT/TDWB/config/BrokerWorkstation.properties`. The default is 31114.

Stopping dynamic workload broker

1. Use **stopBrokerApplication.sh** on UNIX and Linux or **stopBrokerApplication.bat** on Windows as follows:

   stopBrokerApplication -user *username* -password *password* [-port *portnumber*]

   where *username* and *password* are the credentials used during the installation. The parameter *portnumber* is optional. If it is not defined, the default is used.
2. Run the **link** command. If you do not run this command, the dynamic workload broker server is automatically linked ten minutes after the stop operation.

Displaying dynamic workload broker status

Use **brokerApplicationStatus.sh** on UNIX and Linux or **brokerApplicationStatus.bat** on Windows as follows:

brokerApplicationStatus -user *username* -password *password* [-port *portnumber*]

where *username* and *password* are the credentials used during the installation. The parameter *portnumber* is optional. If it is not defined, the default is used.

Automatically initializing IBM Workload Scheduler instances

On UNIX systems, you can automatically initialize IBM Workload Scheduler instances during operating system startup.

About this task

On UNIX systems, you can ensure that your IBM Workload Scheduler instances are automatically initialized during operating system startup.

For AIX, Solaris, HP-UX and some Linux operating systems that use a traditional **init** like**System V**, you can do this by adding an IBM Workload Scheduler service to

the **init** process of your operating system. Use the sample start script **iwa_init**_<*installation user*> located in `TWA_home/TWS/config` and add it to the appropriate run level after customizing it as necessary.

For some Linux distributions that use **systemd** as the default initialization system, such as RedHat Enterprise Linux v7.0 and SUSE Linux Enterprise Server V12, a sample service file, `iwa.service`, is provided located in the path `TWA_home/TWS/config` that is already configured to support the automatic initialization of IBM Workload Scheduler instances at startup.

Perform the following steps:

For UNIX operating systems that use the traditional init system such as `System V` :
Configure the script and then register the service.

1. Create a copy of the **iwa_init**_<`installation user`> script based on your requirements. Provide the following information, depending on the operating system you use:

   Required-Start
   On Linux systems, specify the precondition services

   Default-Start
   On Linux systems, specify the required runlevels. For example, specify runlevels 2, 3, and 5.

2. Browse to the appropriate system-dependent folder, as follows:

   Supported Linux operating systems
   Save the script in the /etc/init.d folder and register the service using the **insserv -v *script_name*** command.

   Supported AIX operating systems
   Save the script to the appropriate rc*runlevel*.d folder. Rename the script according to the runlevel script definition. For example, S*sequence_numberservice_name*, as in S10iwa_init_<`installation user`>.

   Supported Solaris operating systems
   Save the script in the /etc/init.d folder and link the script to an appropriate file in the rc*runlevel*.d folder. For example, S*sequence_numberservice_name*, as in S10tws860ma.

   Supported HP-UX operating systems
   Save the script in the /sbin/init.d folder and link the script to an appropriate file in the rc*runlevel*.d folder. For example, S*sequence_numberservice_name*, as in S10tws860ma.

Note: If you run this script on a master domain manager or on a backup master domain manager, the script has no effect on the database.
For more information about the **inittab, init.d, insserv**, and **init** commands, see the reference documentation for your operating system.

For Linux distributions that use `systemd` as the default initialization system:
This procedure uses the `iwa.service` sample service that is already customized to automatically initialize IBM Workload Scheduler instances at system startup.

1. Copy the sample service provided, `iwa.service`, located in the path `TWA_home/TWS/config` to the following path on the Linux system `/etc/systemd/system/`

2. To make **systemd** aware of the service, invoke the **systemctl daemon-reload** command.
3. Start the service submitting the following command:
   ```
   systemctl start iwa.service
   ```
4. Verify the status by submitting the following command:
   ```
   systemctl status iwa.service
   ```
5. Stop the service by submitting the following command:
   ```
   systemctl stop iwa.service
   ```
6. Finally, enable the service so that it is activated by default when the system boots by submitting the following command:
   ```
   systemctl enable iwa.service
   ```

Application server tasks

The following application server tasks might need to be performed:

Application server - starting and stopping

Use the **startappserver** and **stopappserver** commands or the equivalent from the Dynamic Workload Console to start or stop the WebSphere Application Server. For a description of these commands, see *IBM Workload Scheduler: User's Guide and Reference*.

These commands also stop **appservman**, the service that monitors and optionally restarts the application server.

If you do not want to stop **appservman**, you can issue **startWas** or **stopWas**, supplying the **–direct** argument.

The complete syntax of **startWas** and **stopWas** is as follows:

UNIX

Start the application server

```
startWas.sh  [-direct]
```

Stop the application server

```
stopWas.sh  [-direct]
           [-user <user_ID>
            -password <password>]
```

Note: The above syntax for stopping the WebSphere Application Server is applicable for a master domain manager WebSphere Application Server. If your WebSphere Application Server is for the Dynamic Workload Console, you must use the following syntax:

```
stopWas.sh [-direct]
           [-user <user_ID>
          -password <password>]
```

The user ID and password are optional only if you have specified them in the `soap.client.props` file located in the properties directory of the WebSphere Application Server profile. Unlike the master domain manager installation, when you install the Dynamic Workload Console the `soap.client.props` file is not automatically customized with these credentials.

Windows

Start the application server

```
startWas.bat [-direct]
             [-service <service_name>]
             [-options <parameters>]
```

Stop the application server

```
stopWas.bat [-direct]
            [-service <service_name>]
            [-wasHome <installation_directory>]
            [-user user_ID -password password]
            [-options <parameters>]
```

where the arguments are as follows:

–direct
Optionally starts or stops the application server without starting or stopping the application server monitor **appservman**.

For example, you might use this after changing some configuration parameters. By stopping WebSphere Application Server without stopping **appservman**, the latter will immediately restart WebSphere Application Server, using the new configuration properties. This argument is mandatory on UNIX when the product components are not integrated.

–options *<parameters>*
Optionally supplies parameters to the WebSphere Application Server **startServer** or **stopServer** commands. See the WebSphere Application Server documentation for details.

–password *<password>*
Defines the password to be used when stopping the application server on UNIX. This parameter is optional, if you have set the password in the `soap.client.props` file located in the properties directory of the WebSphere Application Server profile.

–service *<service_name>*
Defines the WebSphere Application Server service name, if it is not the default value of `IBM WebSphere Application Server V6 - <TWS_user>`

–user *<user_ID>*
Defines the user ID to be used when stopping the application server on UNIX. This parameter is optional, if you have set the user ID in the `soap.client.props` file located in the properties directory of the WebSphere Application Server profile.

–wasHome *<installation_directory>*
Defines the WebSphere Application Server installation directory, if it is not the default value.

Application server - automatic restart after failure

If you experience any problems with the application server failing, a service is available that not only monitors its status, but can also restart it automatically in the event of failure. The service is called **appservman**, and it is enabled and controlled by the local options on the computer where the application server is running.

The following sections describe the service, how it works, and how it is controlled:

- "**Appservman** - how it works" on page 449
- "Controlling **appservman**" on page 449

Appservman - how it works

Appservman is a service that starts, stops and monitors the application server. It also optionally restarts the application server in the event that the latter fails. **Appservman** can be controlled not just from nodes running the application server, but also from any other node running **conman**.

It is launched as a service by **netman** when starting IBM Workload Scheduler, and it itself then launches the application server. **Netman** also launches it when the **conman startappserver** command is run.

Appservman is stopped when IBM Workload Scheduler is shut down. In addition, **Netman** stops both the application server and **appservman** when you use the **conman stopappserver** command, or, on Windows only, when you issue the **Shutdown –appsrv** command.

While it is running **appservman** monitors the availability of the application server, sending events that report the status of the application server. If the automatic restart facility is enabled, and the application server fails, the service determines from the restart policy indicated in the `localopts` options if it is to restart the application server. If the policy permits, it will restart the application server, and send events to report its actions.

The WebSphere Application Server utilities **startWas** and **stopWas** by default start and stop the application server and **appservman** using the **startappserver** and **stopappserver** commands. However, these utilities can be instructed to start and stop the application server without stopping **appservman** by using the **startWas** and **stopWas** utilities with the **–direct** option.

Controlling appservman

Appservman is controlled by the following local options (in the `localopts` file):

Appserver auto restart
Determines if the automatic restart facility is enabled.

The default is *yes*. To disable the option set it to *no*.

Appserver check interval
Determines how frequently the service checks on the status of the application server. You should not set this value to less than the typical time it takes to start the application server on the computer.

The default is every 5 minutes.

Appserver min restart time
Determines the minimum time that must elapse between failures of the application server for the automatic restart to work. This option stops **appservman** from immediately restarting the application server if it fails on initial startup or when being restarted.

The default is 10 minutes.

Appserver max restarts
Determines the maximum number of times that **appservman** will

automatically restart the application server within a time frame determined by you (Appserver count reset interval).

The default is 5 restarts.

Appserver count reset interval
Determines the time frame for the maximum number of restarts (Appserver max restarts).

The default is 24 hours.

Appserver service name
This is used in Windows only. It is generated as follows:

`IBMWAS61Service - <TWS_user>`

How to use the options: The default settings are a good starting point. Follow the indications below if you are not satisfied that the settings are maintaining the correct availability of the application server:

- If the application server is not restarting after failure, check the following:
  - That the *Appserver auto restart* is set to *yes*.
  - That the *Appserver check interval* is not set to too high a value. For example, if this value is set to *50* minutes, instead of the default *5*, an early failure of the application server might wait 45 minutes before being restarted.
  - That *Appserver min restart time* is sufficient for the application server to fully restart. If, when the server checks the status of the application server, it finds that the application server is still starting up, in some circumstances it is not able to distinguish the starting-up state from the failed state, will report it as failed, and try and restart it again. With the same result. This will continue until *Appserver max restarts* is exceeded. If this is the case, make *Appserver min restart time* larger.
- If the application server is failing infrequently, but after several failures is not restarting, set the *Appserver max restarts* option to a higher value or the *Appserver count reset interval* to a lower value, or both. In this case it might be advantageous to study the pattern of failures and tailor these options to give you the required availability

Starting and stopping the application server and appservman

If you need to stop and restart the application server, for example to implement a change in the application server configuration, use the following commands:

stopappserver[*domain***!]***workstation* **[;wait]**
This command stops the application server and **appservman**. You can optionally stop the application server on a remote workstation. The optional **;wait** parameter tells **conman** to suspend processing until the command reports that both the application server and the service have stopped.

startappserver[*domain***!]***workstation* **[;wait]**
This command starts the application server and **appservman**. You can optionally start the application server on a remote workstation. The optional **;wait** parameter tells **conman** to suspend processing until the command reports that both the application server and the service are up and running.

To stop and start the application server without stopping **appservman**, see "Application server - starting and stopping" on page 447.

Configuring user and password for running conman stopappserver

When you run **conman stopappserver**, the appserverman process first checks if WebSphere Application Server can retrieve the user's credentials (username and password) from the soap.client.props file located in the WebSphere Application Server profile. If the check is negative, appserverman reads them from the useropts file of the user and runs the stopServer.sh (bat) script to pass them to WebSphere Application Server.

To be able to run **conman stopappserver**, you must therefore complete one of the following two customization procedures to provide the user credentials to WebSphere Application Server:

- Customize the user name (com.ibm.SOAP.loginUserid) and password (com.ibm.SOAP.loginPassword) properties in the soap.client.props file located in:

WAS_profile_path/properties (Version 9.1 and later master and agents)

  where *WAS_profile_path* corresponds to the WebSphere Application Server profile path you specified at installation time. The default value for this path is: *TWA_home*/WAS/TWSprofile.

  You must also:

  1. Set property com.ibm.SOAP.securityEnabled to true in the same file to enable the SOAP client security
  2. Run the encryptProfileProperties.sh script to encrypt the password. See the IBM Workload Scheduler *Administration Guide* for more information on this application server tool.

- Customize the Attributes for conman (CLI in version 8.4) connections section in the localopts file by specifying the details of the connector or of the master domain manager.

  You must also:

  1. Create (or customize if already present) the useropts file manually, adding the USERNAME and PASSWORD attributes for the user who will run **stopappserver**. Make sure the useropts file name is entered in the USEROPTS key in the Attributes for conman (CLI) connections section. See the *Administration Guide* for further details.
  2. Encrypt the password in the useropts file simply by running **conman**.

Monitoring the application server status

To see the current status of the application server at any time view the STATE field in the workstation details.

This field contains a string of characters that provide information about the statuses of objects and processes on the workstation. The state of the application server is a one-character flag in this string, which has one of the following values if the application server is installed:

[A|R]

where:

A The WebSphere Application Server is running.

R The WebSphere Application Server is restarting.

If the application server is down or if it was not installed, neither value of the flag is present in the STATE entry.

Obtaining information about application server failures

Appservman does not provide information about why the application server has failed. To obtain this information, look in the application server log files (see the *IBM Workload Scheduler: Troubleshooting Guide*).

Events created by appservman

Appservman sends an event called *ApplicationServerStatusChanged* from the `TWSObjectsMonitor` provider to the configured event monitoring process every time the status of the application server changes.

Application server - encrypting the profile properties files

You use the **encryptProfileProperties** script to encrypt passwords in the following WebSphere Application Server property files:

- *<WAS_profile_path>*`/properties/soap.client.props`
- *<WAS_profile_path>*`/properties/sas.client.props`
- *<WAS_profile_path>*`/properties/sas.stdclient.properties`
- *<WAS_profile_path>*`/properties/sas.tools.properties`

where *WAS_profile_path* corresponds to the WebSphere Application Server profile path you specified when you installed one of the following components: master domain manager, backup master domain manager, dynamic domain manager, backup dynamic domain managers. The default path is: *TWA_home*`/WAS/TWSprofile`.

An example of when you might use the encryption function is when creating SSL key-stores. You would type the passwords in the key-stores and then encrypt them using the **encryptProfileProperties** script. See *IBM Workload Scheduler: Planning and Installation*.

The script uses **PropFilePasswordEncoder.bat**. Restart the server for the changes to take effect.

Encrypt properties uses the following syntax: "Application server - encrypting the profile properties files"

```
encryptProfileProperties.bat (.sh)
```

Application server - updating the Windows services after modifications

If you have modified any of the following, you must update the Windows service that runs the application server:

- The user ID and password of the local operating system user that runs the application server process
- The installation directory of the WebSphere Application Server
- The directory where the IBM Workload Scheduler application server profile is stored

To update the service, run the **updateWasService** command from the *<TWA_home>*`/wastools` directory.

Note: You can also use this command to change the way that the application server is started.

At run time, the script calls **WASService.exe**.

updateWasService
Format

updateWasService –userid <*TWS_user*> –password <*TWS_user_password*>
[–wasuser <WAS_user> –waspassword <WAS_user_password>]
[–startType {automatic | manual | disabled}]
[–wasHome <WebSphere_install_directory>]
[–profilePath <server_profile_directory>]

Parameters

–userid <*TWS_user*> –password <*TWS_user_password*>
Supply the <*TWS_user*> and its password.

[–wasuser <WAS_user> –waspassword <WAS_user_password>]
The user that the local operating system uses to run the application process is set by default to the <*TWS_user*>. If you want to change it to a different user and password, supply this parameter.

Note: Due to a known problem with this utility, when you change the password you must first use the Windows facility for changing the password, as described in "Action 7 - Windows - update Windows services" on page 427, and then run this utility, giving the new password you have just set as the <*WAS_user_password*>.

[–startType {automatic | manual | disabled}]
The application server starts automatically by default, when the computer is started. If you want to have a different behavior supply this parameter.

[–wasHome <WebSphere_install_directory>]
If you have changed the name of the installation directory of the WebSphere Application Server, supply this parameter indicating the new name.

[–profilePath <server_profile_directory>]
If you have changed the name of the directory where the IBM Workload Scheduler application server profile is stored, supply this parameter indicating the new name.

Application server - updating the SOAP properties after changing the WebSphere Application Server user or its password

If you have changed the user ID or the password of the WebSphere Application Server administration user either for IBM Workload Scheduler or the Dynamic Workload Console, you must also update the SOAP client properties.

To update the properties, run the **updateWas.sh/.bat** command from the <*TWA_home*>/wastools directory.

After using this command you must restart the application server.

updateWas.sh
Format

updateWas.sh –user <*new_WAS_admin_user*> –password <*password*>

Parameters

–user *<new_WAS_admin_user>* **–password** *<password>*
Supply the user and password of the new WebSphere Application Server administration user that you want to be configured as the credentials in the SOAP client properties.

Application server - configuration files backup and restore

The application server has configuration files, which should be backed up whenever you change them. Use the **backupConfig** script in the *<TWA_home>*/wastools directory for IBM Workload Scheduler or in the *<TDWC_INSTALL_PATH>*/wastools for the Dynamic Workload Console.

The files are restored, if necessary, using the **restoreConfig** script in the same directory.

There is no need to stop the application server to perform the backup, but you must stop and restart the server if you have restored the files from a previous backup.

For further information, see the *IBM Redbooks: WebSphere Application Server V6 System Management & Configuration Handbook*.

Backup usage

Backup uses the following syntax:

```
backupConfig.bat [backup_file]
                 [-nostop]
                 [-quiet]
                 [-logfile file_name]
                 [-replacelog]
                 [-trace]
                 [-username user_ID]
                 [-password password]
                 [-profileName profile]
                 [-help]
```

where,

***backup_file* for the Dynamic Workload Console**
is the file (full name and path) used to back up the JazzSM profile configuration. If *backup_file* is not specified, then the output is saved by default in a compressed file as follows:
DynamicWorkloadConsole_installpath/TDWC/backup/WebSphereConfig_backup.zip

***backup_file* for IBM Workload Scheduler**
is the file (full name and path) used to back up the JazzSM profile configuration. If *backup_file* is not specified, then the command does not run.

The following is an example of **backupconfig.bat**:

```
C:\Program Files\ibm\TWA0\wastools>backupConfig.bat
ADMU0116I: Tool information is being logged in file
           C:\Program Files\ibm\TWA0\WAS\TWSprofile\logs\
           backupConfig.log
ADMU0128I: Starting tool with the twsprofile profile
ADMU5001I: Backing up config directory
           C:\Program Files\ibm\TWA0\WAS\TWSprofile\config to file
           C:\Program Files\ibm\TWA0\wastools\WebSphereConfig_2005-12-12.zip
ADMU0505I: Servers found in configuration:
```

```
ADMU0506I: Server name: server1
ADMU2010I: Stopping all server processes for node DefaultNode
ADMU0512I: Server server1 cannot be reached. It appears to be stopped.
.....................................................................
ADMU5002I: 137 files successfully backed up
```

Restore usage

Restore uses the following syntax issued from the *profile_root*/bin directory:

```
restoreConfig.bat [backup_file]
                  [-location restore_location]
                  [-quiet]
                  [-nowait]
                  [-logfile file_name]
                  [-replacelog]
                  [-trace]
                  [-username user_ID]
                  [-password password]
                  [-profileName profile]
                  [-help]
```

where,

backup_file **for the Dynamic Workload Console**
: is the file (full name and path) used to restore the JazzSM profile configuration. If *backup_file* is not specified, then the backup file named *DynamicWorkloadConsole_installpath*/TDWC/backup/WebSphereConfig_backup.zip is used if it exists.

backup_file **for IBM Workload Scheduler**
: is the file (full name and path) used to restore the JazzSM profile configuration. If *backup_file* is not specified, then the command does not run.

The following is an example of **restoreConfig.bat**:

```
C:\Program Files\ibm\TWA0\wastools>restoreConfig.bat WebSphereConfig_2005-12-11.zip
ADMU0116I: Tool information is being logged in file
           C:\Program Files\ibm\TWA0\WAS\TWSprofile\logs\
           restoreConfig.log
ADMU0128I: Starting tool with the twsprofile profile
ADMU0505I: Servers found in configuration:
ADMU0506I: Server name: server1
ADMU2010I: Stopping all server processes for node DefaultNode
ADMU0512I: Server server1 cannot be reached. It appears to be stopped.
ADMU5502I: The directory C:\Program Files\ibm\TWA0\WAS\TWSprofile\config
           already exists; renaming to
           C:\Program Files\ibm\TWA0\WAS\TWSprofile\config.old
ADMU5504I: Restore location successfully renamed
ADMU5505I: Restoring file WebSphereConfig_2005-12-11.zip to location
           C:\Program Files\ibm\TWA0\WAS\TWSprofile\config
.....................................................................
ADMU5506I: 127 files successfully restored
ADMU6001I: Begin App Preparation -
ADMU6009I: Processing complete.
```

Application server - changing the host name or TCP/IP ports

To modify the host name of the computer where the application server is installed, or the TCP/IP ports it uses, run the **changeHostProperties** script.

The procedure requires you to stop the application server, create a text file of the current host properties, edit the file, run the utility and restart the application server. Find information about how to do this as follows:

- "Application server - using the utilities that change the properties" on page 459 gives a generic description of the procedure for making any change to the WebSphere Application Server properties
- "Changing host properties" lists all the host properties and gives other reference information about the utility
- When editing the text file of the current host properties, do as follows:

  1. Edit the text file and locate the following properties:

```
#################################################################
# Host Configuration Panel
#################################################################

# Old Hostname
oldHostname=myoldhost.mydomain.com

# New Hostname
newHostname=myhost.mydomain.com

#################################################################
Ports Configuration Panel
#################################################################
bootPort=41117
bootHost=myhost.mydomain.com
soapPort=41118
soapHost=myhost.mydomain.com
httpPort=41115
httpHost=*
httpsPort=41116
httpsHost=*
adminPort=41123
adminHost=*
adminSecurePort=41124
adminSecureHost=*
sasPort=41119
sasHost=myhost.mydomain.com
csiServerAuthPort=41120
csiServerAuthHost=myhost.mydomain.com
csiMuthualAuthPort=41121
csiMuthualAuthHost=myhost.mydomain.com
orbPort=41122
orbHost=myhost.mydomain.com
```

     The rules for changing these values are as follows:

     – To change the host name, supply both `oldHostname` and `newHostname`. Also check that the values of `bootHost` and `csiServerAuthHost` are set correctly (normally to the new host name).
     – If you change the host name, the old host port settings are used, unless you specifically change them.
     – If you do not supply a port number it is not changed.

     Do not change any other properties.

Note: The utility might display a message from the application server (WASX7357I:). You can ignore this message.

Changing host properties

You use the **changeHostProperties** script to change the workstation host name in the WebSphere Application Server configuration files, or the TCP/IP ports used by WebSphere Application Server. To change or disable TCP/IP Ports see "Disabling TCP/IP ports" on page 458.

The procedure for running the script is described in detail in "Application server - using the utilities that change the properties" on page 459, but in summary you do the following:

- Run **showHostProperties.sh (.bat) > my_file_name** to obtain the current properties
- Edit *my_file_name*
- Run **changeHostProperties.sh (.bat) my_file_name**

Note: *my_file_name* must be the fully qualified path of the file.

The change utility calls the **wsadmin** utility by running **ChangeHostProperties.jacl** with the properties file as input.

Only the WebSphere Application Server `serverindex.xml` configuration file is affected by this script. The path of the file is:

WAS_profile_path`/config/cells/TWSNodeCell/nodes/TWSNode/servers/server1/server.xml`

where the default value of <*WAS_profile_path*> is `<TWA_home>/WAS/TWSprofile` for the master domain manager, and <*JazzSM_profile_dir*> for the z/OS connector and Dynamic Workload Console where, the default value of <*JazzSM_profile_dir*> is `/opt/IBM/JazzSM/profile`.

The following is a list of the properties that can be changed with this utility:

```
################################################################
# Host Configuration Panel
################################################################

# Old Hostname
oldHostname=myoldhost.romelab.ibm.it.com

# New Hostname
newHostname=mynewhost.romelab.ibm.it.com

################################################################
Ports Configuration Panel
################################################################
bootPort=41117
bootHost=myhost.mydomain.com
soapPort=41118
soapHost=myhost.mydomain.com
httpPort=41115
httpHost=*
httpsPort=41116
httpsHost=*
adminPort=41123
adminHost=*
adminSecurePort=41124
adminSecureHost=*
sasPort=41119
sasHost=myhost.mydomain.com
csiServerAuthPort=41120
csiServerAuthHost=myhost.mydomain.com
csiMuthualAuthPort=41121
csiMuthualAuthHost=myhost.mydomain.com
orbPort=41122
orbHost=myhost.mydomain.com
```

All of the properties in the properties file are optional. When you are modifying the properties file you should be aware of the following:

- When **oldHostname** and **newHostname** are provided (these settings must be provided together), the host property related to each port is updated when it has not been provided in the properties file and the current value matches **oldHostname**.
- Port settings are updated only if they are provided in the properties file.
- A port-specific host setting, such as **httpHost** is not updated if not specified except when its current setting matches **oldHostname**.
- An empty setting is considered as not provided.

Disabling TCP/IP ports: Using the **changeHostProperties** script you can also disable some TCP/IP ports by setting the value of the following corresponding properties to **false**. If you are using Secure Sockets Layer Communication (SSL) in your network, disabling the non-secure HTTP and Administrative Console ports will ensure that only encrypted communication occurs in your network. By default these ports are all enabled. To disable them use the following properties:

httpEnabled
To disable the `httpPort`.

httpsEnabled
To disable the `httpsPort`.

adminEnabled
To disable the `adminPort`.

adminSecureEnabled
To disable the `adminSecurePort`.

Displaying the current host properties: To display the current properties, use the following utility:

UNIX **showHostProperties.sh**

Windows
showHostProperties.bat

Application server - changing the trace properties

You can use the **changeTraceProperties** script to change the WebSphere Application Server trace properties.

The script calls the **wsadmin** utility by running the **ChangeServerTracing.jacl** with a properties file whose template is `TracingProps.properties`.

See the *IBM Workload Scheduler: Troubleshooting Guide* for full details on how to change the trace settings.

The properties file defines the following trace modes:

```
wsmm_odr=com.ibm.ws.xd.comm.*=all:com.ibm.wsmm.grm.Controller=
all:com.ibm.ws.xd.work*
=all:com.ibm.ws.xd.arfm.*=all:com.ibm.wsmm.policing.*
=all:com.ibm.wsmm.xdglue.*
=all:com.ibm.ws.odc.ODCTreeImpl$Save=all
wsmm_node=com.ibm.ws.xd.comm.*=all:com.ibm.ws.xd.placement*
=all:com.ibm.ws.xd.arfm.*=all
reset=*=info
wsmm007=com.ibm.wsmm.policing.*=all
dcs=DCS=finest:RMM=finest
ham=hamanageditem=all
tcpdcs=DCS=finest:RMM=finest:com.ibm.ws.tcp.channel.*=finest
tcp=com.ibm.ws.tcp.channel.*=finest
```

```
vizcache=com.ibm.ws.xd.visualizationengine.cacheservice.cacheimpl.*=all
runtime=com.ibm.ws.console.xdruntime.*=all
proxy=com.ibm.ws.console.proxy.*=all
placement=com.ibm.ws.xd.placement*=all=enabled
charting=com.ibm.ws.console.chart.*=all
dwlm=com.ibm.ws.dwlm.*=all
operationalpolicy=com.ibm.ws.xd.operationalpolicymonitor.*=all
wsmm_na=*=info:com.ibm.ws.xd.comm.*=all:com.ibm.ws.xd.placement*
=all:com.ibm.ws.xd.workprofiler.*=all:com.ibm.ws.xd.arfm.*=
all:com.ibm.ws.dwlm.*
=all:com.ibm.ws.xd.hmm.*=all:com.ibm.ws.xd.admin.utils.*
=all:com.ibm.ws.clustersensor.impl.*
=all:com.ibm.ws.xd.placement.memory.profiler.impl.*=off
wsmm_o=*=info:com.ibm.ws.xd.comm.*=all:com.ibm.wsmm.grm.Controller=
all:com.ibm.ws.xd.workprofiler.*
=all:com.ibm.ws.xd.arfm.*=all:com.ibm.wsmm.policing.*=
all:com.ibm.wsmm.xdglue.*
=all:com.ibm.ws.dwlm.*=all:com.ibm.ws.dwlm.client.*=off
dmgr=com.ibm.ws.odc.*=
all:com.ibm.ws.xd.visualizationengine.cacheservice.cacheimpl.
DeploymentTargetCache*
=all
grid=grid.capacityplacement=all
webcontainer=com.ibm.ws.webcontainer.*=all:com.ibm.ws.http.*=all
odc=com.ibm.ws.odc.*=all:com.ibm.ws.dwlm.client.*=
all:com.ibm.ws.xd.dwlm.client.*
=all:com.ibm.ws.proxy.*=all
wssec_all=com.ibm.ws.security.*=all
wssec_tws_all=com.ibm.ws.security.*=all:com.ibm.tws.*=all
tws_all=com.ibm.tws.*=all
tws_alldefault=com.ibm.tws.*=error=enabled
tws_db=com.ibm.tws.dao.model.*=all:com.ibm.tws.dao.rdbms.*=all
tws_planner=com.ibm.tws.planner.*=all:com.tivoli.icalendar.*=
all:com.ibm.tws.runcycles.*
=all:com.ibm.tws.conn.planner.*=all:com.ibm.tws.cli.planner.*=all
tws_cli=com.ibm.tws.cli.*=all:com.ibm.tws.objects.*=all
tws_utils=com.ibm.tws.util.*=all
tws_conn=com.ibm.tws.conn.*=all:com.ibm.tws.objects.*=
all:com.ibm.tws.updatemanager.*
=all:com.ibm.tws.dao.plan.*=all
tws_secjni=com.ibm.tws.audit.*=all:com.ibm.tws.security.*=all
active_correlation=com.ibm.correlation.*=all
tws_jni=TWSJNI=all
tws_all_jni=com.ibm.tws.*=all:TWSJNI=all
tws_all_act=com.ibm.tws.*=all:com.ibm.correlation.*=all
tws_broker_all=com.ibm.scheduling.*=all:TWSAgent=all
tws_broker_rest=com.ibm.scheduling.jobmanager.rest.*=all
tws_engine_broker_all=com.ibm.tws.*=all:com.ibm.scheduling.*=
all:TWSAgent=all
tws_bridge=TWSAgent=all
tws_db_transactions=com.ibm.tws.planner.currentplan.PlannerEngine=
all:com.ibm.tws.dao.rdbms.util.DatabaseTransaction=all
```

You can define other trace modes and update `TracingProps.properties` or create a new properties file.

Two settings that must not be changed are the server name (default **server1**), and the node (default **DefaultNode**).

WebSphere Application Server tools - reference

Application server - using the utilities that change the properties

This section documents a common procedure that you are advised to follow when using the following utilities:

- changeDataSourceProperties
- changeHostProperties
- changeSecurityProperties

To avoid the risk of changing a configuration value inadvertently, you should follow a procedure that creates a file containing the current properties, edit it to the values you require, and apply the changes. The details are as follows:

1. Log on to the computer where IBM Workload Scheduler is installed as the following user:

   UNIX root

   Windows
   Any user in the *Administrators* group.
2. Access the directory: `<TWA_home>/wastools`
3. Stop the WebSphere Application Server using the **conman stopappserver** command (see "Starting and stopping the application server and **appservman**" on page 450)
4. From that same directory run the following script to create a file containing the current properties:

   UNIX **show<property_type>Properties.sh > my_file_name**

   Windows
   show<property_type>Properties.bat > my_file_name

   where <*property_type*> is one of the following:
   - DataSource
   - Host
   - Security
5. Edit `my_file_name` with a text editor. Check the start of the file. The command might have written a message from the application server (WASX7357I:) at the beginning of the file. Delete this message.
6. Change the value of the configuration parameters, according to your requirements. You do not need to supply all of the parameters in the file.
7. Save the file *my_file_name*.
8. Run the script:

   Windows
   change<property_type>Properties.bat my_file_name

   UNIX **change<property_type>Properties.sh my_file_name**

   where <*property_type*> is the same as used in step 4, and *my_file_name* is the *fully qualified path* of the file containing the new parameters.

   The properties are updated, according to the rules given in the descriptions of each property type.
9. Start the WebSphere Application Server using the **conman startappserver** command (see "Starting and stopping the application server and **appservman**" on page 450)
10. Check that the change has been implemented in IBM Workload Scheduler.

Understanding the templates

As indicated in the overview to these utilities, a template file of properties is supplied with the product for each of these utilities. However, this template file

does not contain any of the configuration values that were created by the product installation process or any previous uses of the configuration utilities. If you decide to use the template instead of creating the properties document as described in step 4 on page 460, you must be careful to ensure that the values you enter in the file for every parameter are those that you require to be used by the application server.

Application server utilities

This section provides reference information about the utilities provided with the WebSphere Application Server that supports IBM Workload Scheduler.

The utilities are a set of scripts based on Windows batch files, UNIX and Linux shell scripts, and WebSphere Jacl procedures. The scripts run the WebSphere Application Server utilities that perform the reconfiguration. Many of them load configuration settings from a properties file. Templates for these files are also provided.

IBM Workload Scheduler installs the following Windows scripts:

- backupConfig.bat
- brokerApplicationStatus.bat
- changeBrokerSecurityProperties.bat
- changeDataSourceProperties.bat
- changeHostProperties.bat
- changePassword.bat
- changeSecurityProperties.bat
- changeTraceProperties.bat
- encryptProfileProperties.bat
- InstallOracledataSource.bat
- manage_ltpa.bat
- modifyThreadPool.bat
- restoreConfig.bat
- setEnv.bat
- showBrokerSecurityProperties.bat
- showDataSourceProperties.bat
- showHostProperties.bat
- showSecurityProperties.bat
- startBrokerApplication.bat
- startWas.bat
- stopBrokerApplication.bat
- stopWas.bat
- updateWas.bat
- updateWasService.bat

IBM Workload Scheduler installs the following UNIX and Linux scripts:

- backupConfig.sh
- brokerApplicationStatus.sh
- changeBrokerSecurityProperties.bat
- changeDataSourceProperties.sh
- changeHostProperties.sh

- changePassword.sh
- changeSecurityProperties.sh
- changeTraceProperties.sh
- createCustomRegistryforPAM.sh
- encryptProfileProperties.sh
- InstallOracledataSource.sh
- manage_ltpa.sh
- modifyThreadPool.sh
- restoreConfig.sh
- setEnv.sh
- showBrokerSecurityProperties.sh
- showDataSourceProperties.sh
- showHostProperties.sh
- showSecurityProperties.sh
- startBrokerApplication.sh
- startWas.sh
- stopBrokerApplication.sh
- stopWas.sh
- manage_ltpa.sh
- modifyThreadPool.sh
- InstallOracledataSource.sh
- updateWas.sh
- wasstart.sh

The following templates are installed for both Windows and UNIX operating systems:

- BrokerSecurityProps.properties
- DataSourceProps.properties
- HostConfigProps.properties
- SecurityProps_FULL.properties
- SecurityProps_TEMPLATE.properties
- TracingProps.properties

See “Application server - using the utilities that change the properties” on page 459 to understand how the templates should be used.

Chapter 10. Administering an IBM i dynamic environment

On overview on how to administer the IBM Workload Scheduler IBM i dynamic environment.

To begin scheduling jobs with advanced options on IBM i agents, the agents must be configured.

Configuring the agent on IBM i systems

An overview on how to configure the agent on IBM i systems.

The configuration settings of the agent are contained in the `JobManager.ini` file and in the `JobManagerGW.ini` file (for the path of these files, see "Where products and components are installed" on page 1).

The configuration files are made up of many different sections. Each section name is enclosed between square brackets and each section includes a sequence of `variable = value` statements.

You can customize properties for the following:

- Log properties
- Trace properties when the agent is stopped. You can also customize traces when the agent is running using the procedure described in "Configuring trace properties when the agent is running" on page 61.
- Native job executor
- Java job executor
- Resource advisor agent
- System scanner

On IBM i systems, the log messages are written in the following file:

```
<TWA_home>/TWS/stdlist/JM/JObManager_message.log
```

On IBM i systems, the trace messages are written in the following files:

```
<TWA_home>/TWS/stdlist/JM/ITA_trace.log
<TWA_home>/TWS/stdlist/JM/JobManager_trace.log
<TWA_home>/TWS/stdlist/JM/javaExecutor0.log
```

Not all the properties in the `JobManager.ini` file and in the `JobManagerGW.ini` file can be customized. For a list of the configurable properties, see the following sections:

- "Configuring log message properties [JobManager.Logging.cclog]" on page 59.
- "Configuring trace properties when the agent is stopped [JobManager.Logging.cclog]" on page 60.
- "Configuring common launchers properties [Launchers]" on page 64.
- "Configuring properties of the native job launcher [NativeJobLauncher]" on page 65.
- "Configuring properties of the Java job launcher [JavaJobLauncher]" on page 68.
- "Configuring properties of the Resource advisor agent [ResourceAdvisorAgent]" on page 68.

- "Configuring properties of the System scanner [SystemScanner]" on page 70

Note: In the `JobManager.ini` file and in the `JobManagerGW.ini` file you must refer to Java 64 bit version.

Configuring log message properties [JobManager.Logging.cclog]

About this task

To configure the logs, edit the [JobManager.Logging.cclog] section in the `JobManager.ini` file. This procedure requires that you stop and restart the IBM Workload Scheduler agent

The section containing the log properties is named:
`[JobManager.Logging.cclog]`

You can change the following properties:

JobManager.loggerhd.fileName
The name of the file where messages are to be logged.

JobManager.loggerhd.maxFileBytes
The maximum size that the log file can reach. The default is 1024000 bytes.

JobManager.loggerhd.maxFiles
The maximum number of log files that can be stored. The default is 3.

JobManager.loggerhd.fileEncoding
By default, log files for the agent are coded in UTF-8 format. If you want to produce the log in a different format, add this property and specify the required codepage.

JobManager.loggerfl.level
The amount of information to be provided in the logs. The value ranges from 3000 to 7000. Smaller numbers correspond to more detailed logs. The default is 3000.

JobManager.ffdc.maxDiskSpace
Exceeding this maximum disk space, log files collected by the first failure data capture mechanism are removed, beginning with the oldest files first.

JobManager.ffdc.baseDir
The directory to which log and trace files collected by the ffdc tool are copied. Default directory is `<TWA_home>\TWS\stdlist\JM\JOBMANAGER-FFDC`.

JobManager.ffdc.filesToCopy
Log and trace files (JobManager_message.log and JobManager_trace.log) collected by the ffdc tool located in `<TWA_home>\TWS\stdlist\JM`. For example, `JobManager.ffdc.filesToCopy = "/opt/IBM/TWA_<TWS_user>/TWS/stdlist/JM/JobManager_message.log" "/opt/IBM/TWA_<TWS_user>/TWS/stdlist/JM/JobManager_trace.log"`

When a message is logged (JobManager.ffdc.triggerFilter = JobManager.msgIdFilter) that has an ID that matches the pattern "AWSITA*E" (JobManager.msgIdFilter.msgIds = AWSITA*E), which corresponds to all error messages, then the log and trace files (JobManager.ffdc.filesToCopy = "/opt/IBM/TWA_<TWS_user>/TWS/stdlist/JM/JobManager_message.log" "/opt/IBM/TWA_<TWS_user>/TWS/stdlist/JM/JobManager_trace.log") are copied

(JobManager.ffdc.className = ccg_ffdc_filecopy_handler) to the directory JOBMANAGER-FFDC (JobManager.ffdc.baseDir = /opt/IBM/TWA_<TWS_user>/TWS/stdlist/JM/JOBMANAGER-FFDC). If the files copied exceed 10 MB (JobManager.ffdc.maxDiskSpace = 10000000), then the oldest files are removed first (JobManager.ffdc.quotaPolicy = QUOTA_AUTODELETE).

After installing the z-centric agent or dynamic agent on Windows 2012, the JobManager_message.log might not be created. In this case, perform the following procedure:

1. Stop the agent.
2. Create a backup copy of JobManager.ini, and edit the original file by changing the row:

   ```
   JobManager.loggerhd.className = ccg_multiproc_filehandler
   ```

   to

   ```
   JobManager.loggerhd.className = ccg_filehandler
   ```
3. Restart the agent.

Configuring trace properties when the agent is stopped [JobManager.Logging.cclog]

How to configure the trace properties when the agent is stopped.

To configure the trace properties when the agent is stopped, edit the [JobManager.Logging] section in the JobManager.ini file and then restart the IBM Workload Scheduler agent.

The section containing the trace properties is named:

```
[JobManager.Logging.cclog]
```

You can change the following properties:

JobManager.trhd.fileName
The name of the trace file.

JobManager.trhd.maxFileBytes
The maximum size that the trace file can reach. The default is 1024000 bytes.

JobManager.trhd.maxFiles
The maximum number of trace files that can be stored. The default is 3.

JobManager.trfl.level
Determines the type of trace messages that are logged. Change this value to trace more or fewer events, as appropriate, or on request from IBM Software Support. Valid values are:

DEBUG_MAX
Maximum tracing. Every trace message in the code is written to the trace logs.

INFO All *informational*, *warning*, *error* and *critical* trace messages are written to the trace. The default value.

WARNING
All *warning*, *error* and *critical* trace messages are written to the trace.

ERROR
All *error* and *critical* trace messages are written to the trace.

CRITICAL
Only messages which cause the agent to stop are written to the trace.

The output trace (JobManager_trace.log) is provided in XML format.

* After installing the z-centric agent or dynamic agent on Windows 2012, the
* `JobManager_trace.log` might not be created. In this case, perform the following
* procedure:
1. Stop the agent.
2. Create a backup copy of `JobManager.ini`, and edit the original file by changing the row:

   ```
   JobManager.trhd.className = ccg_multiproc_filehandler
   ```

   to

   ```
   JobManager.trhd.className = ccg_filehandler
   ```
3. Restart the agent.

Configuring trace properties when the agent is running

Use the **twstrace** command to set the trace on the agent when it is running.

Using the **twstrace** command, you can perform the following actions on the agent when it is running:

- "See command usage and verify version" on page 62.
- "Enable or disable trace" on page 62.
- Set the traces to a specific level, specify the number of trace files you want to create, and the maximum size of each trace file. See "Set trace information" on page 62.
- "Show trace information" on page 63.
- Collect trace files, message files, and configuration files in a compressed file using the command line. See "Collect trace information" on page 63.
- Collect trace files, message files, and configuration files in a compressed file using the Dynamic Workload Console. See the section about retrieving IBM Workload Scheduler agent traces from the Dynamic Workload Console in *Troubleshooting Guide*.

You can also configure the traces when the agent is not running by editing the [JobManager.Logging] section in the `JobManager.ini` file as described in Configuring the agent section. This procedure requires that you stop and restart the agent.

twstrace command

Use the **twstrace** command to configure traces, and collect logs, traces, and configuration files (ita.ini and jobManager.ini) for agents. You collect all the information in a compressed file when it is running without stopping and restarting it.

See command usage and verify version

To see the command usage and options, use the following syntax.

Syntax

twstrace -u | -v

Parameters

-u Shows the command usage.

-v Shows the command version.

Enable or disable trace

To set the trace to the maximum or minimum level, use the following syntax.

Syntax

twstrace -enable | -disable

Parameters

-enable
Sets the trace to the maximum level. The maximum level is **1000**.

-disable
Sets the trace to the minimum level. The minimum level is **3000**.

Set trace information

To set the trace to a specific level, specify the number of trace files you want to create, and the maximum size the trace files can reach, use the following syntax.

Syntax

twstrace [**-level** <*level_number*>] [**-maxFiles** <*files_number*>] [**-maxFileBytes** <*bytes_number*>]

Parameters

-level <*level_number*>
Sets the trace level. Specify a value in the range from 1000 to 3000, which is also the default value. Note that if you set this parameter to 3000, you have the lowest verbosity level and the fewest trace messages. To have a better trace level, with the most verbose trace messages and the maximum trace level, set it to **1000**.

-maxFiles <*files_number*>
Specify the number of trace files you want to create.

-maxFileBytes <*bytes_number*>
Set the maximum size in bytes that the trace files can reach. The default is **1024000** bytes.

Show trace information

To display the current trace level, the number of trace files, and the maximum size the trace files can reach, use the following syntax.

Syntax

twstrace -level | **-maxFiles** | **-maxFileBytes**

Parameters

-level
See the trace level you set.

-maxFiles
See the number of trace files you create.

-maxFileBytes
See the maximum size you set for each trace file

Sample

The example shows the information you receive when you run the following command:

```
twstrace -level -maxFiles -maxFileBytes
```

```
AWSITA176I The trace properties are: level="1000",
max files="3", file size="1024000".
```

Collect trace information

To collect the trace files, the message files, and the configuration files in a compressed file, use the following syntax.

Syntax

twstrace -getLogs [**-zipFile** <*compressed_file_name*>] [**-host** <*host_name*>] [**-protocol** {*http* | *https* } [**-port** <*port_number*>][**-iniFile** <*ini_file_name*>]

Parameters

-zipFile <*compressed_file_name*>
Specify the name of the compressed file that contains all the information, that is logs, traces, and configuration files (ita.ini and jobManager.ini) for the agent. The default is **logs.zip**.

-host <*host_name*>
Specify the host name or the IP address of the agent for which you want to collect the trace. The default is **localhost**.

-protocol *http*|*https*
Specify the protocol of the agent for which you are collecting the trace. The default is the protocol specified in the **.ini** file of the agent.

-port <*port_number*>
Specify the port of the agent. The default is the port number of the agent where you are running the command line.

-iniFile <*ini_file_name*>
Specify the name of the **.ini** file that contains the SSL configuration of the agent for which you want to collect the traces. If you are collecting the traces for a remote agent for which you customized the security certificates, you must import the certificate on the local agent and specify the name of the **.ini** file that contains this configuration. To do this, perform the following actions:

1. Extract the certificate from the keystore of the remote agent.

2. Import the certificate in a local agent keystore. You can create an ad hoc keystore whose name must be **TWSClientKeyStore.kdb**.
3. Create an **.ini** file in which you specify:
   - **0** in the **tcp_port** property as follows:
     `tcp_port=0`
   - The port of the remote agent in the **ssl_port** property as follows:
     `ssl_port=<ssl_port>`
   - The path to the keystore you created in Step 2 on page 64 in the **key_repository_path** property as follows:
     `key_repository_path=<local_agent_keystore_path>`

Configuring common launchers properties [Launchers]

About this task

In the `JobManager.ini` file, the section containing the properties common to the different launchers (or executors) is named:

`[Launchers]`

You can change the following properties:

BaseDir
The installation path of the IBM Workload Scheduler agent.

CommandHandlerMinThreads
The default is **20**.

CommandHandlerMaxThreads
The default is **100**.

CpaHeartBeatTimeSeconds
The polling interval in seconds used to verify if the **agent** process is still up and running. If the agent process is inactive the product stops also the **JobManager** process. The default is **30**.

DirectoryPermissions
The access rights assigned to the agent for creating directories when running jobs. The default is **0755**. Supported values are UNIX-format entries in hexadecimal notation.

DownloadDir
The name of the directory where the fix pack installation package or upgrade eImage for fault-tolerant agents or dynamic agents is downloaded during the centralized agent update process. If not specified, the following default directory is used:

On Windows operating systems:
`<TWA_home>\TWS\stdlist\JM\download`

On UNIX operating systems:
`<TWA_home>/TWS/stdlist/JM/download`

The centralized agent update process doesn't apply to z-centric agents.

ExecutorsMaxThreads
The default is **400**.

ExecutorsMinThreads
The default is **38**.

FilePermissions
The access rights assigned to the agent for creating files when running jobs. The default is **0755**. Supported values are UNIX-format entries in hexadecimal notation.

MaxAge
The number of days that job logs are kept (in path *TWA_home*/TWS/stdlidst/JM) before being deleted. The default is **30**. Possible values range from a minimum of 1 day.

NotifierMaxThreads
The default is **5**.

NotifierMinThreads
The default is **3**.

SpoolDir
The path to the folder containing the jobstore and outputs. The default is:
value of BaseDir/stdlidst/JM

StackSizeBytes
The size of the operating system stack in bytes. The default is **DEFAULT**, meaning that the **agent** uses the default value for the operating system.

Configuring properties of the native job launcher [NativeJobLauncher]

About this task

In the JobManager.ini file, the section containing the properties of the native job launcher is named:
[NativeJobLauncher]

You can change the following properties:

AllowRoot
Applies to UNIX systems only. Specifies if the root user can run jobs on the agent. It can be true or false. The default is false. This property does not apply to IBM i.

CheckExec
If true, before launching the job, the agent checks both the availability and the execution rights of the binary file. The default is true.

DefaultWorkingDir
Specifies the working directory of native jobs. You can also specify the value for the working directory when creating or editing the job definition in the Workload Designer. When specified in the Workload Designer, this value overrides the value specified for the **DefaultWorkingDir** property. If you do not specify any working directories, the <*TWS_home*>\bin directory is used.

JobUnspecifiedInteractive
Applies to Windows operating systems only. Specifies if native jobs are to be launched in interactive mode. It can be true or false. The default is false.

KeepCommandTraces
Set to true to store the traces of the method invocation for actions performed on a job definition, for example, when selecting from a picklist.

These files are stored in the path /opt/IBM/TWA_<TWS_user>/TWS/stdlist/JM/r3batch_cmd_exec. The default setting is false.

KeepJobCommandTraces
Set to true to store the traces of the method invocation for actions performed on a job instance, for example, viewing a spool list. These files are stored in the .zip file of the job instance. The default setting is true.

LoadProfile
/ Applies to agents on Windows servers only. Specifies if the user profile is to be loaded. It can be true or false. The default is true.

MonitorQueueName
Specifies the name of the queue where the IBM i jobs are monitored. If you do not specify this property, the default queue (QBATCH) is used.

PortMax
The maximum range of the port numbers used by the task launcher to communicate with the Job Manager. The default is 0, meaning that the operating system assigns the port automatically.

PortMin
The minimum range of the port numbers used by the task launcher to communicate with the Job Manager. The default is 0, meaning that the operating system assigns the port automatically.

PostJobExecScriptPathName
The fully qualified path of the script file that you want to run when the job completes. By default, this property is not present in the JobManager.ini file. If you do not specify any file path or the script file doesn't exist, no action is taken.

This property applies to dynamic agent and z/OS agent. For details about running a script when a job completes, see *User's Guide and Reference*.

PromotedNice
Used in workload service assurance. This property is not supported on the Agent for z/OS.

For UNIX and Linux operating systems only, assigns the priority value to a critical job that needs to be promoted so that the operating system processes it before others. Applies to critical jobs or predecessors that need to be promoted so that they can start at their critical start time.

Boundary values vary depending upon each specific platform, but generally lower values correspond to higher priority levels and vice versa. The default is -1.

Be aware that:

- The promotion process is effective with negative values only. If you set a positive value, the system runs it with the -1 default value.
- An out of range value (for example -200), prompts the operating system to automatically promote the jobs with the lowest allowed nice value.
- Overusing the promotion mechanism (that is, defining an exceedingly high number of jobs as mission critical and setting the highest priority value here) might overload the operating system, negatively impacting the overall performance of the workstation.

PromotedPriority
Used in workload service assurance. This property is not supported on the Agent for z/OS.

For Windows operating systems only, sets to this value the priority by which the operating system processes a critical job when it is promoted. Applies to critical jobs or predecessors that need to be promoted so that they can start at their critical start time. Valid values are:

- `High`
- `AboveNormal` (the default)
- `Normal`
- `BelowNormal`
- `Low` or `Idle`

Note that if you a set a lower priority value than the one non-critical jobs might be assigned, no warning is given.

RequireUserName

When `true`, requires that you add the user name in the JSDL job definition.

When `false`, runs with the user name used by job manager, that is:

- *TWS_user* on UNIX and Linux systems
- The local system account on Windows systems

The default is `false`.

RunExecutablesAsIBMiJobs

If you set this property to `true`, you can define IBM i jobs as generic jobs without using the XML definition. Generic jobs are automatically converted to IBM i jobs. As a side effect, generic jobs cannot be run when this parameter is enabled (`RunExecutablesAsIBMiJobs=true`). There is no default value because this property is not listed in the `JobManager.ini` file after the agent installation.

If you set this property to `true`, ensure that the user you used to install the agent has been granted the `*ALLOBJ` special authority.

ScriptSuffix

The suffix to be used when creating the script files. It is:

`.cmd` For Windows

`.sh` For UNIX

VerboseTracing

Enables verbose tracing. It is set to `true` by default.

Configuring properties of the Java job launcher [JavaJobLauncher]

About this task

In the `JobManager.ini` file, the section containing the properties of the Java job launcher is named:

`[JavaJobLauncher]`

You can change the following properties:

JVMDir

The path to the virtual machine used to start job types with advanced options. You can change the path to another compatible Java virtual machine.

JVMOptions

The options to provide to the Java Virtual Machine used to start job types with advanced options. Supported keywords for establishing a secure connection are:

- htttps.proxyHost
- https.proxyPort

Supported keywords for establishing a non-secure connection are:

- Dhttp.proxyHost
- Dhttp.proxyPort

For example, to set job types with advanced options, based on the default JVM http protocol handler, to the unauthenticated proxy server called with name myproxyserver.mycompany.com, define the following option:

```
JVMOptions = -Dhttp.proxyHost=myproxyserver.mycompany.com
 -Dhttp.proxyPort=80
```

Configuring properties of the Resource advisor agent [ResourceAdvisorAgent]

About this task

In the `JobManager.ini` and `JobManagerGW.ini` files, the section containing the properties of the Resource advisor agent is named:

```
[ResourceAdvisorAgent]
```

You can change the following properties:

BackupResourceAdvisorUrls

The list of URLs returned by the IBM Workload Scheduler master in a distributed environment or by the dynamic domain manager either in a z/OS or in a distributed environment. The agent uses this list to connect to the master or dynamic domain manager.

CPUScannerPeriodSeconds

The time interval that the Resource advisor agent collects resource information about the local CPU. The default value is every 10 seconds.

FullyQualifiedHostname

The fully qualified host name of the agent. It is configured automatically at installation time and is used to connect with the master in a distributed environment or with the dynamic domain manager in a z/OS or in a distributed environment. Edit only if the host name is changed after installation.

NotifyToResourceAdvisorPeriodSeconds

The time interval that the Resource advisor agent forwards the collected resource information to the Resource advisor. The default (and maximum value) is every 180 seconds.

ResourceAdvisorUrl

JobManager.ini

The URL of the master in a distributed environment, or of the dynamic domain manager in a z/OS or in a distributed environment, that is hosting the agent. This URL is used until the server replies with the list of its URLs. The value is `https://$(tdwb_server):$(tdwb_port)/JobManagerRESTWeb/JobScheduler/resource`, where:

$(*tdwb_server***)**
is the fully qualified host name of the master in a distributed environment or of the dynamic domain manager either in a z/OS or in a distributed environment.

$(*tdwb_port***)**
is the port number of the master in a distributed environment or of the dynamic domain manager either in a z/OS or in a distributed environment.

It is configured automatically at installation time. Edit only if the host name or the port number are changed after installation, or if you do not use secure connection (set to `http`). If you set the port number to zero, the resource advisor agent does not start. The port is set to zero if at installation time you specify that you will not be using the master in a distributed environment or the dynamic domain manager either in a z/OS or in a distributed environment.

In a distributed environment, if **-gateway** is set to either `local` or `remote`, then this is the URL of the dynamic agent workstation where the gateway resides and through which the dynamic agents communicate. The value is `https://$(tdwb_server):$(tdwb_port)/ita/JobManagerGW/JobManagerRESTWeb/JobScheduler/resource`, where:

$(*tdwb_server***)**
The fully qualified host name of the dynamic agent workstation where the gateway resides and through which the dynamic agent communicates with the dynamic workload broker.

$(*tdwb_port***)**
The port number of the dynamic agent workstation where the gateway resides and through which the dynamic agent communicates with the dynamic workload broker.

JobManagerGW.ini
In a distributed environment, if **-gateway** is set to `local`, then **ResourceAdvisorUrl** is the URL of the master or dynamic domain manager. The value is `https://$(tdwb_server):$(tdwb_port)/JobManagerRESTWeb/JobScheduler/resource`, where:

$(*tdwb_server***)**
The fully qualified host name of the master or dynamic domain manager.

$(*tdwb_port***)**
The port number of the master or dynamic domain manager.

ScannerPeriodSeconds
The time interval that the Resource advisor agent collects information about all the resources in the local system other than CPU resources. The default value is every 120 seconds.

The resource advisor agent, intermittently scans the resources of the machine (computer system, operating system, file systems and networks) and periodically sends an update of their status to the master or dynamic domain manager either in a z/OS or in a distributed environment.

The CPU is scanned every `CPUScannerPeriodSeconds` seconds, while all the other resources are scanned every `ScannerPeriodSeconds` seconds. As soon as one of the scans shows a significant change in the status of a resource, the resources are synchronized with the master in a distributed environment or the dynamic domain manager either in a z/OS or in a distributed environment. The following is the policy followed by the agent to tell if a resource attribute has significantly changed:

- A resource is added or deleted
- A string attribute changes its value
- A CPU value changes by more than `DeltaForCPU`
- A file system value changes by more than `DeltaForDiskMB` megabytes
- A Memory value changes by more than `DeltaForMemoryMB` megabytes

If there are no significant changes, the resources are synchronized with the IBM Workload Scheduler master in a distributed environment or with thedynamic domain manager either in a z/OS or in a distributed environment every `NotifyToResourceAdvisorPeriodSeconds` seconds.

Configuring properties of the System scanner [SystemScanner]

About this task

In the `JobManager.ini` file, the section containing the properties of the System scanner is named:

`[SystemScanner]`

You can change the following properties:

CPUSamples
The number of samples used to calculate the average CPU usage. The default value is 3.

DeltaForCPU
The change in CPU usage considered to be significant when it becomes higher than this percentage (for example, DeltaForCPU is 20 if the CPU usage changes from 10 percent to 30 percent). The default value is 20 percent.

DeltaForDiskMB
The change in use of all file system resources that is considered significant when it becomes higher than this value. The default value is 100 MB.

DeltaForMemoryMB
The change in use of all system memory that is considered significant when it becomes higher than this value. The default value is 100 MB.

Configuring to schedule job types with advanced options

About this task

In addition to defining job types with advanced options using the Dynamic Workload Console or the **composer** command, you can use the related configuration files. The options you define in the configuration files apply to all job types with advanced options of the same type. You can override these options when defining the job using the Dynamic Workload Console or the **composer** command.

Configuration files are available on each dynamic agent in TWA_home/TWS/JavaExt/cfg for the following job types with advanced options:

Table 87. Configuration files for job types with advanced options

Job type	File name	Keyword
• Database job type • MSSQL Job	DatabaseJobExecutor.properties	Use the `jdbcDriversPath` keyword to specify the path to the JDBC drivers. Define the keyword so that it points to the JDBC jar files directory, for example: `jdbcDriversPath=c:\\mydir\\jars\\jdbc` The JDBC jar files must be located in the specified directory or its subdirectories. Ensure you have list permissions on the directory and its sub subdirectories. **Note:** For the MSSQL database, use version 4 of the JDBC drivers.
Java job type	JavaJobExecutor.properties	Use the `jarPath` keyword to specify the path to the directory where the jar files are stored. This includes all jar files stored in the specified directory and all sub directories.
J2EE job type	J2EEJobExecutorConfig.properties	For more information about the J2EE job type, see Configuring to schedule J2EE jobs.

Customizing the SSL connection between IBM i agents and a master domain manager or a dynamic domain manager using your own certificates

Customizing the SSL connection between a master domain manager or a dynamic domain manager and IBM i agents connected to it using your own certificates.

About this task

By default the communication between IBM i agents and a master domain manager or a dynamic domain manager to which they are registered uses the https protocol.

The SSL communication uses the default certificates provided by IBM Workload Scheduler.

If you want to use your own customized certificates for this communication because you customized the master domain manager or the dynamic domain manager certificates, you must customize the agent certificates and the agent configuration file.

To enable communication between a master domain manager or a dynamic domain manager and an IBM i agent, you must first create your own certificates for IBM i agent and then trust the agents certificates in the master domain manager or the dynamic domain manager keystore.

Perform the following steps:

1. Log on as Administrator on Windows operating systems or as root on UNIX and Linux operating systems, on the machine where you installed a IBM Workload Scheduler instance that contains the **openssl** utility, for example, the master domain manager or the dynamic domain manager.
2. Go to the <*TWS_INST_DIR*>/TWS/ssl directory, where <*TWS_INST_DIR*> is the IBM Workload Scheduler installation directory and copy there the following files:
   - <*TWS_INST_DIR*>/TWS/bin/openssl(.exe)
   - <*TWS_INST_DIR*>/TWS/bin/openssl.cnf
3. Generate a random file for the IBM i agent, by using the following command:

   ```
   openssl rand
   -out <suffix>.rnd
   -rand ./openssl 8192
   ```

   where <*suffix*> is a generic word. For example, you can use the IBM i agent workstation name to easily find the files generated for this workstation.
4. Generate the <*suffix*>.key private key, by running the following command:

   ```
   openssl genrsa -des3
   -out <suffix>.key 2048
   ```

   and save the password that you entered in the previous command in the <*suffix*>.pwd file.

   Note: Ensure that you take note of the password you insert because you need it in the following steps.
5. Generate the ita_prv<*suffix*>.pem PEM file containing the agent private key, by renaming the <*suffix*>.key in ita_prv<*suffix*>.pem.
6. Save the agent private key password in a <*suffix*>.sth stash file by using the following command:

   ```
   openssl base64
   -in <suffix>.pwd
   -out <suffix>.sth
   ```
7. Generate the <*suffix*>.csr certificate signature request by running the following command:

   ```
   openssl req -new
   -key <suffix>.key
   -out <suffix>.csr
   -config ./openssl.cnf
   ```
8. Generate the <*suffix*>.crt certificate that contains the private key <*suffix*>.key by running the following command:

   ```
   openssl x509 -req
   -CA TWSca.crt
   -CAkey TWSca.key
   ```

```
-days 365
-in <suffix>.csr
-out <suffix>.crt
-CAcreateserial
```

9. Generate the <*suffix*>.pem PEM file containing the agent private key certificate by creating a copy of the <*suffix*>.crt certificate, and name the copied file <*suffix*>.pem.
10. Generate the ita_pub<*suffix*>.pem PEM file containing the agent private key certificate by creating a copy of the <*suffix*>.crt certificate, and name the copied file ita_pub<*suffix*>.pem.
11. Create a copy of the ita_pub<*suffix*>.pem file created in step 10 and name the copied file ita_cert<*suffix*>.pem.
12. On the master domain manager or the dynamic domain manager machine to which the IBM i agent is to be connected, generate the server.pem certificate by running the command:

    ```
    keytool -export -rfc
    -alias server
    -file <TWS_INST_DIR>/TWS/ssl/server.pem
    -keypass <password>
    -keystore <TWS_INST_DIR>/TWA/WAS/TWSprofile/etc/TWSServerKeyFile.jks
    -storepass default
    ```

    where <password> is the value you entered in step 4 on page 477.
13. Generate the ita_ca_cert<*suffix*>.pem file which is the concatenation of the ita_pub<*suffix*>.pem and of the server.pem files, by performing the following actions:
    a. Create a copy of the ita_pub<*suffix*>.pem file and name it ita_ca_cert<*suffix*>.pem.
    b. Edit the ita_ca_cert<*suffix*>.pem file.
    c. Append at the end of the ita_ca_cert<*suffix*>.pem file content the server.pem file content.
    d. Save the final version of the ita_ca_cert<*suffix*>.pem file.

    Note: The ita_ca_cert<*suffix*>.pem file contains the certificates of the IBM i agent and the master domain manager or the dynamic domain manager to which the agent is connected.
14. Log on as <*TWS_IBMi_USER*> user on the IBM i agent machine and locate the <*TWS_IBMI_INSTDIR*>/TWS/ITA/cpa/ita/cert/ directory where <*TWS_IBMI_INSTDIR*> is the directory where you installed the IBM Workload Scheduler IBM i agent for the <*TWS_IBMi_USER*> user.
15. From the <*TWS_INST_DIR*>/TWS/ssl directory of the machine where you generated the PEM files, copy into the <*TWS_IBMI_INSTDIR*>/TWS/ITA/cpa/ita/cert/ directory of the IBM i agent installation directory the following files:
    - ita_prv<*suffix*>.pem.
    - ita_pub<*suffix*>.pem.
    - ita_cert<*suffix*>.pem.
    - ita_ca_cert<*suffix*>.pem.
    - <*suffix*>.sth.
    - <*suffix*>.rnd.

    Note: Ensure that the files you copied have <*TWS_IBMi_USER*> ownership.

16. On the machine where you installed the IBM i agent, open the ita.ini configuration agent file and set the values appropriate for your environment in the following properties:

    password_file=<stash_file_fullpath>
    random_file=<random_file_fullpath>
    cert_label=<label_agent_private_key>
    key_db_name=<*suffix*>
    key_repository_dir=<directory_ita_*<*suffix*>.pem>

    Where:

    <stash_file_fullpath>
    Specify the fully qualified path to the <*suffix*>.sth stash file that contains the agent private key password. This is the file you created in step 6 on page 477. The default value is <*TWS_IBMI_INSTDIR*>/TWS/ITA/cpa/ita/cert/password.sth.

    <random_file_fullpath>
    Specify the fully qualified path to the <*suffix*>.rnd random file. This is the file that you created in step 3 on page 477. The default is <*TWS_IBMI_INSTDIR*>/TWS/ITA/cpa/ita/cert/TWS.rnd.

    <label_agent_private_key>
    Specify the label of the agent private key.

    <*suffix*>
    Specify the suffix that you used in the names of all the files that you generated. The default value is **tws**.

    <directory_ita_*<*suffix*>.pem>
    Specify the directory that contains the following .pem files that you generated:

    Truststore
    ita_ca_cert<*suffix*>.pem that you generate in step 13 on page 478

    Keystore
    - ita_prv<*suffix*>.pem that you generated in step 5 on page 477.
    - ita_pub<*suffix*>.pem that you generated in step 10 on page 478.
    - ita_cert<*suffix*>.pemthat you generated in step 11 on page 478.

    The default directory is <*TWS_IBMI_INSTDIR*>/TWS/ITA/cpa/ita/cert.

17. Stop the IBM i agent by using the following command:

    ShutDownLwa

18. Start the IBM i agent by using the following command:

    StartUpLwa

19. On the master domain manager or the dynamic domain manager machine which the IBM i agent is to be connected to, trust the <*TWS_INST_DIR*>/TWS/ssl/<*suffix*>.pem IBM i agent certificate that you generated in step 9 on page 478, in the keystore, by running the following steps:

    keytool -import -trustcacerts
    -alias <*suffix*>
    -file <*TWS_INST_DIR*>/TWS/ssl/<*suffix*>.pem

```
-keypass <password>
-keystore <TWS_INST_DIR>/TWA/WAS/TWSprofiles/etc/
    TWSServerTrustFile.jks
-storepass default
```

where <*TWS_INST_DIR*> is the master domain manager or the dynamic domain manager installation directory and <password> is the value you entered in step 4 on page 477.

Example

You have the following environment:

- IBM i agent installed in the opt/ibm/TWS directory of the nc117031 machine for the user twsuserIBMi.
- Master domain manager installed in the opt/IBM/TWA92 directory of the machine nc060201.

To create the IBM i agent certificates to connect to the master domain manager, perform the following steps:

1. Log on as root on the nc060201 machine where you installed the master domain manager.
2. Go to the opt/IBM/TWA92/TWS/ssl directory and copy there the following files:
   - opt/IBM/TWA92/TWS/bin/openssl
   - opt/IBM/TWA92/TWS/bin/openssl.cnf
3. Generate the nc117031.rnd random file in the opt/IBM/TWA92/TWS/ssl directory by running the following command:

   ```
   openssl rand
   -out nc117031.rnd
   -rand ./openssl 8192
   ```
4. Generate the nc117031.key private key in the opt/IBM/TWA92/TWS/ssl directory by running the following command:

   ```
   openssl genrsa -des3
   -out nc117031.key 2048
   ```

   and save the maestro00 password that you entered in the nc117031.pwd file in text format in the opt/IBM/TWA92/TWS/ssl directory.
5. Create a copy of the nc117031.key file in the opt/IBM/TWA892/TWS/ssl directory and name it ita_prvnc117031.pem.
6. Save the maestro00 password in a nc117031.sth stash file in the opt/IBM/TWA92/TWS/ssl directory by running the following command:

   ```
   openssl base64
   -in nc117031.pwd
   -out nc117031.sth
   ```
7. Generate the nc117031.csr certificate signature request in the opt/IBM/TWA92/TWS/ssl directory by running the following command:

   ```
   openssl req -new
   -key nc117031.key
   -out nc117031.csr
   -config ./openssl.cnf
   ```
8. Generate the nc117031.crt certificate in the opt/IBM/TWA92/TWS/ssl directory that contains the private key nc117031.key by running the following command:

```
openssl x509 -req
-CA TWSca.crt
-CAkey TWSca.key
-days 365
-in nc117031.csr
-out nc117031.crt
-CAcreateserial
```

9. Create a copy of the nc117031.crt certificate in the opt/IBM/TWA92/TWS/ssl directory and name it nc117031.pem.
10. Create a copy of the nc117031.crt certificate in the opt/IBM/TWA92/TWS/ssl directory and name it ita_pubnc117031.pem.
11. Create a copy of the ita_pubnc117031.pem file in the opt/IBM/TWA92/TWS/ssl directory and name it ita_certnc117031.pem.
12. On the nc060201 machine, generate the server.pem certificate in the opt/IBM/TWA92/TWS/ssl directory by running the following command:

    ```
    keytool -export -rfc
    -alias server
    -file opt/IBM/TWA/TWS/ssl/server.pem
    -keypass maestro00
    -keystore opt/IBM/TWA/WAS/TWSprofile/etc/TWSServerKeyFile.jks
    -storepass default
    ```

13. Generate the ita_ca_certnc117031.pem file in the opt/IBM/TWA/TWS/ssl directory which is the concatenation of the ita_pubnc117031.pem and the server.pem files, by performing the following actions:
    a. Create a copy of ita_pubnc117031.pem file in the opt/IBM/TWA/TWS/ssl directory and name it ita_ca_certnc117031.pem.
    b. Edit the ita_ca_certnc117031.pem file.
    c. Append at the end of the ita_ca_certnc117031.pem file content the server.pem file content.
    d. Save the final version of the ita_ca_certnc117031.pem file.
14. Log on as twsuserIBMi user on the nc117031 machine and locate the opt /ibm/TWS/ITA/cpa/ita/cert/directory.
15. From the opt/IBM/TWA/TWS/ssl directory of the nc060201 machine where you generated the PEM files, copy into the opt/ ibm/TWS/ITA/cpa/ita/cert/ directory the following files:
    - ita_prvnc117031.pem.
    - ita_pubnc117031.pem.
    - ita_certnc117031.pem.
    - ita_ca_certnc117031.pem.
    - nc117031.sth.
    - nc117031.rnd.

    Ensure that all the files have twsuserIBMi ownership.
16. On the nc117031 machine, open the ita.ini configuration agent file and set the following values for the listed properties:

    ```
    password_file=opt/ibm/TWS/ITA/cpa/ita/cert/nc117031.sth
    random_file=opt/ibm/TWS/ITA/cpa/ita/cert/nc117031.rnd
    cert_label=nc117031
    key_db_name=nc117031
    key_repository_dir=opt/ibm/TWS/ITA/cpa/ita/cert/*nc117031.pem>
    ```

17. Stop the IBM i agent by using the following command:

    ```
    ShutDownLwa
    ```

18. Start the IBM i agent by using the following command:

```
StartUpLwa
```

19. On the nc060201 machine, trust the opt/IBM/TWA92/TWS/ssl/nc117031.pem agent certificate by running the following steps:

```
keytool -import -trustcacerts
-alias nc117031
-file opt/IBM/TWA/TWS/ssl/ssl/nc117031.pem
-keypass maestro00
-keystore opt/IBM/TWA/WAS/TWSprofile/etc/TWSServerTrustFile.jks
-storepass default
```

Chapter 11. Performance

This chapter provides information about issues that impact performances. Use this information both to prevent problems occurring and to help resolve problems that have occurred.

Network traffic

A full description of how a IBM Workload Scheduler network is structured, and how the different nodes communicate, is provided at the beginning of Chapter 6, "Network administration," on page 271. In particular, see "Optimizing the network" on page 285, which explains how to design and operate your IBM Workload Scheduler network to maximize performance.

Tracing

The performance of any workstation can be impacted by the level of tracing it has to perform. The IBM Workload Scheduler Troubleshooting Guide has a chapter which explains the diagnostic tools that are available, and within that chapter there is a section about the IBM Workload Scheduler In-flight Tracing utility, which, as well as discussing how the feature works, also describes how to customize it to enhance workstation performance.

The performance might also be impacted by the tracing activities on the WebSphere Application Server.

Logging

The performance of any workstation can be impacted by the way the IBM Workload Scheduler logging mechanism uses memory. The default settings applied in this version are designed to ensure the maximum performance. However, because these defaults are different from the defaults in earlier versions, if you are experiencing performance problems, it is advisable to check that these settings have not been in some way overwritten by the previous values. In the diagnostic tools chapter of *IBM Workload Scheduler: Troubleshooting Guide*, there is a section about CCLog, which, apart from discussing how to customize CCLog, also describes how to check the CCLog processing defaults.

Maintaining the database

Maintaining the database in a good state of organization is important to optimize performance. See "Reorganizing the database" on page 357 for details.

Symphony file sizing

To calculate the size of the Symphony file and understand its impact on performance, see "Avoiding full file systems" on page 358.

Tuning a UNIX domain manager to handle large numbers of fault-tolerant agents

The performance of domain managers on UNIX is impacted if they serve large numbers of fault-tolerant agents. Improvements can be obtained by modifying the kernel parameters. The precise settings differ according to operating system, and you might need to test different settings to obtain optimum performance.

The following is an example of the kernel settings for HP-UX to handle approximately 200 fault-tolerant agents:

```
max_thread_proc=256
nprocess=1800
maxusers=120
maxuprc=1700
nflocks=500
maxfiles=1024
```

Tuning job processing on a workstation

This section explains how to tune selected options in the IBM Workload Scheduler `localopts` file to improve IBM Workload Scheduler performance. These options control the period between successive instances of an activity. Table 88 shows the activities to be tuned, the corresponding option that can be set in the `localopts` file, and how the changed value impacts performance.

Table 88. Options for tuning job processing on a workstation

Activity	Option	Impact on performance
batchman periodically scans the `Symphony` file for jobs ready to be processed.	***bm look***	In all these cases, a shorter time means more frequent scans, using more cpu resources, and impacting other processes that are running. However, it also means that for all activities waiting time is kept to a minimum. If throughput is important and the workstation has plenty of memory, try shortening the times. A longer period between successive activities means jobs take longer to run, because there are longer waits for each activity. However, the reduced frequency of the scans means that more memory is available for jobs because less is being used by these monitoring activities. Consider the meaning of the various options. If your objective is to run the jobs as quickly as possible, but you are not concerned about how quickly the information about completed jobs is distributed, you could reduce the wait periods for *bm look* and *jm read*, but increase the periods for the others. Alternatively, to speed up the overall job processing time (from initial job launch to the update with the completion status), you can tune *bm look*, *jm look*, and *mm read*.
jobman accesses the `Courier.msg` file to see if there are jobs that need to be launched.	***jm read***	
After having launched a job **jobman** checks periodically for job completion status.	***jm look***	
mailman looks periodically in the `Mailbox.msg` for completed jobs.	***mm read***	
batchman checks periodically in `Intercom.msg` for jobs that are complete so that it can update the `Symphony` file.	***bm read***	

If you decide to tune these setting do the following:

- Test the result in a test system before applying changes in your production environment. To get worthwhile results, the test environment must have the same characteristics as the production environment.
- Modify only the parameters that are necessary. It is better to modify one at a time and thoroughly test the change in performance, rather than changing all at once.
- Make a backup copy of the `localopts` file to ensure you can revert to the default options if necessary.

Stop and start the agent to activate changes applied to the `localopts` file.

Tuning plan replication

Tuning plan replication involves configuring specific settings to optimize the process of replicating plan data into the database. Plan replication ensures quick and reliable access to plan data stored in the database. Its main objective is to provide quick response times and increased overall performance. Sometimes, if this synchronization process is not configured appropriately for the size of your workload, you might notice some discrepancies in your environment, such as job status misalignment between the command line (**conman**) and the monitoring results obtained in the Dynamic Workload Console.

There are a few simple settings you can configure to optimize performance:

Configure the cache size

Add the following properties to the `TWSConfig.properties` file located in *<TWS_INSTALLATION_PATH>*`/WAS/TWSProfile/properties/TWSConfig.properties`:

```
#Custom property which configures the mirroring cache size for file
 dependencies (default value is 10000)
com.ibm.tws.planner.monitor.filecachesize=20000

#Customer property which configures the mirroring cache size
 (default value is 10000)
com.ibm.tws.planner.monitor.cachesize=20000
```

In addition, follow the steps to increase the heap size settings (initialHeapSize = 2048 and maximumHeapSize = 4096) of the application server on the master domain manager as documented in the *Administration Guide*.

Set the statement cache size

Set the WebSphere Application Server data source property **Statement cache size** to `400`. You can make this change from the WebSphere Administrative Console by editing the WebSphere Application Server data source properties page. To access the administrative console page click: **Resources** > **JDBC** > **JDBC providers** > **<JDBC_provider_name>** > **Data sources** > **<data_source_name>** > **WebSphere Application Server data source properties**.

You can also make this change by directly editing the file, *<TWS_INSTALLATION_PATH>*`/WAS/TWSProfile/config/cells/TWSNodeCell/nodes/TWSNode/servers/server1/resources.xml` as follows:

```
<factories xmi:type="resources.jdbc:DataSource" xmi:id=
 "DataSource_1268732051783"
name="DB2 Universal Type 4 JDBC Driver DataSource" jndiName="jdbc/twsdb"
description="DB2 Universal Driver Datasource" providerType="DB2 Universal
 JDBC Driver Provider"
authMechanismPreference="BASIC_PASSWORD" authDataAlias="twsj2c"
```

```
 manageCachedHandles="false"
logMissingTransactionContext="true" diagnoseConnectionUsage="false"
relationalResourceAdapter="builtin_rra" statementCacheSize="400"
datasourceHelperClassname=
 "com.ibm.websphere.rsadapter.DB2UniversalDataStoreHelper">
```

Set min and max connections for data source connection pools

Set the WebSphere Application Server data source connection pools min and max connections to 0 and 300, respectively. You can make this change from the WebSphere Administrative Console by editing the WebSphere Application Server data source properties page. To access the administrative console page click: **Resources** > **JDBC** > **JDBC providers** > **<JDBC_provider_name>** > **Data sources** > **<data_source_name>** > **connection pools**.

You can also make this change by directly editing the file, *<TWS_INSTALLATION_PATH>*/WAS/TWSProfile/config/cells/TWSNodeCell/nodes/TWSNode/servers/server1/resources.xml, as follows:

```
<connectionPool xmi:id="ConnectionPool_1268732051796" connectionTimeout="180"
maxConnections="300"minConnections="0" reapTime="180" unusedTimeout="1800"
agedTimeout="0" purgePolicy="EntirePool" numberOfSharedPoolPartitions="0"
numberOfUnsharedPoolPartitions="0" numberOfFreePoolPartitions="0"
 freePoolDistributionTableSize="0"
surgeThreshold="-1" surgeCreationInterval="0" testConnection="false"
 testConnectionInterval="15"
stuckTimerTime="0" stuckTime="0" stuckThreshold="0"/>
```

Configure the maximum number of threads for DefaultWorkManager Work manager

Set the maximum number of threads equal to 50 for the Work manager named, **DefaultWorkManager**. You can make this change from the WebSphere Administrative Console by editing the WebSphere Application Server DefaultWorkManager Work manager page. To access the administrative console page click: **Resources** > **Asynchronous beans** > **Work managers** > **DefaultWorkManager**.

You can also make this change by directly editing the file, <TWS_INSTALLATION_PATH>/WAS/TWSProfile/config/cells/TWSNodeCell/nodes/TWSNode/servers/server1/resources-pme.xml, as follows:

```
<factories xmi:type="workmanager:WorkManagerInfo" xmi:id="WorkManagerInfo_Default"
name="DefaultWorkManager" jndiName="wm/default"
 description="WebSphere Default WorkManager"
category="Default" minThreads="1" maxThreads="50"
 threadPriority="5" numAlarmThreads="5"
isGrowable="true" workReqQFullAction="0">
```

Install DB2 JDBC Driver Type 4

Replicating plan data in the database requires that DB2 JDBC Driver Type 4 is installed. DB2 JDBC Driver Type 2 is not supported.

Tuning the database

To learn about tuning the database, consult the relevant product documentation:

DB2 Go to IBM DB2 Database for Linux, UNIX, and Windows Information Center, and search for **Best practices**.

Oracle See the *Performance Tuning Guide* in the Oracle documentation set.

Optimizing the replication of the Symphony file in the database

Tuning DB2 database configuration parameters to improve performance when the Symphony file is replicated in the database.

In a IBM Workload Scheduler environment where more than 200,000 jobs are scheduled to be submitted, there are several DB2 database configuration parameters than can be tuned to improve performance when the `Symphony` plan is replicated in the IBM Workload Scheduler database.

The following are the suggested values for a plan with more than 200,000 jobs:

```
*  LOGBUFSZ = 2150
*  DBHEAP = AUTOMATIC (or greater than LOGBUFSZ)
*
*  LOGFILSIZ = 3000
*  LOGPRIMARY = 200
*  LOGSECOND = 40
*
*  PAGE_AGE_TRGT_MCR = 120
```

In addition, increase the number of pages (NPAGES) of the **`TWS_PLN_BUFFPOOL`** parameter to 182000 using the **`ALTER BUFFERPOOL`** command.

Before changing any of these values, refer to the information about tuning a DB2 database in the relevant product documentation at IBM DB2 Database for Linux, UNIX, and Windows Information Center.

Tuning the WebSphere Application Server

To learn about tuning the WebSphere Application Server, consult the appropriate documentation.

Go to http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp, select **WebSphere Application Server (Distributed operating systems), Version 8.5** and then **Tuning performance**.

Inadequate Java heap size

The default Java maximum heap size might be too small for your requirements. If you have any reason to suspect the performance of the WebSphere Application Server, increase the heap size as described in “Increasing application server heap size” on page 492.

Too many manual job submissions

IBM Workload Scheduler is designed for maximum efficiency when handling jobs submitted using a scheduled plan. Consequently, it is less adapted to processing manually submitted jobs. Thus, performance can be improved by reducing the number of manually submitted jobs.

Too many file dependency checks

Each file dependency check has an impact on performance. If you design a plan that is constantly checking many file dependencies, you reduce the performance of the workstation where these jobs are being run.

If multiple "opens" files are being used as a dependency, use the "–a" (and) option. For example, to check if three home directories `/tom`, `/dick`, and `/harry` exist, before launching `myjob` issue the following:

```
job2 opens "/users" (-d %p/tom -a -d %p/dick -a -d %p/harry)
```

This checks for all three directories at the same time, instead of looking for each directory separately.

Other factors that impact performance when evaluating file dependencies are the **bm check** parameters in the `localopts` file. These are documented in the IBM Workload Scheduler: Planning and Installation in the chapter about customization.

Workload spreading

Whatever jobs you have to schedule, try and spread them out through the production period so that there is no concentration in any one moment. Try also to avoid scheduling activities during times when normal user traffic in the network is very heavy, for example during the morning when users commence working and deal with accumulated emails.

Failure to do this might cause a bottleneck at the Mailbox.msg queue, which causes delays in updating the Symphony file, which in turn creates delays in the availability of job statuses to **conman**, the Dynamic Workload Console.

Improving job-processing performance

The processing and monitoring of jobs on a workstation is controlled primarily by various parameters in the `localopts` file and the global options maintained by **optman**. These parameters are described in the *IBM Workload Scheduler: Planning and Installation Guide*.

If you are experiencing problems of performance when processing and monitoring jobs, contact IBM Software Support for advice about how to tune these parameters in your particular environment to improve performance.

Mailbox caching - advantages and disadvantages

Mailman uses a parameter in the `localopts` file to decide whether to cache mailbox messages: *mm cache mailbox*. This section explains the advantages and disadvantages of the on and off settings of this parameter.

Setting the *mm cache mailbox* **parameter to** *no*
: This means that mailman has to make a separate read action for each message before processing it, and then a separate delete action after successfully processing the message. The I/O activity in performing these activities one message at a time is proportionally high for the amount of data being read. This has an impact on performance. On the other hand, the processing is simple, in that each message is read, processed, and then removed from the mailbox. Any failure of the system at any point means that at most one message is replayed and no data is lost.

Setting the *mm cache mailbox* **parameter to** *yes* **(default)**
: This means that mailman reads a block of messages into cache memory, processes all of the messages, and then deletes all of them from the mailbox. The advantage in I/O time is clear; reading and deleting a

sequential set of messages in one action is a much more efficient use of I/O time, than reading and deleting them one-by-one, meaning improved performance.

However, if there is a failure of mailman or the operating system, the cache is lost. On restarting, mailman rereads the set of messages that were previously in cache, some of which might already have been processed. For example, if mailman reads a block of 32 messages into cache and has processed 30 of them when a problem occurs, when mailman is restarted it rereads those 32 records and has to process 30 duplicates before being able to continue where it stopped.

Most events deal with job state changes, and these events can be repeated without creating any problems, and the critical events mechanism is able to deal with the others. However, there is an impact on performance while this recovery processing is going on, and if the in-built mechanisms cannot handle the message duplication, a more serious error might occur, ultimately involving the full or partial loss of the mailbox contents.

The number of messages being read in one action is configurable, using the parameter *mm cache size*. The default value for this parameter is 32 messages, and the maximum is 512. Setting this parameter to a value higher than the default increases performance during correct working, but decreases the performance in the event of a failure, for the reasons stated above. In addition, the additional cache means that the memory required by the IBM Workload Scheduler engine also increases. If you have a workstation with limited memory, or memory-heavy applications running, it might be counterproductive to increase the mailbox cache because the operating system might have to start paging the cache memory.

In conclusion, the default setting maximizes performance; only if you start losing events should you set it to *no*.

Setting the synch level parameter

This section describes the impact of the different settings of the *synch level* parameter in the `localopts` file. The *synch level* parameter only impacts UNIX environments.

The I/O activity performed by the IBM Workload Scheduler engine in managing plans, job streams, and jobs, consists in reading from and writing to the `Symphony` file and the event files (`Mailbox.msg`, `Intercom.msg`, and `Courier.msg`). When IBM Workload Scheduler writes to these files it has more than a straightforward *write* operation to perform. For example, when it writes to the `Mailbox.msg` file it performs the actions described in the following pseudo code:

```
TWS_write_event_lock {
     Lock Mailbox to write
}

TWS_write_event_update {
     Check Available Space
     Write Header
     Write Record
     Update Write Pointer
     Unlock Mailbox
}
```

Each action requires one or more write accesses to the disk. The way these actions are performed with the different synch level options is as follows:

synch level = high
Each write operation on the event files is immediately physically written to disk. This has a heavy impact on performance caused by the high I/O dependency.

synch level = medium
Each write event is considered as a single operation. For example, while `TWS_write_event_lock` contains only one action, `TWS_write_event_update` comprises five actions. With `synch level` at *medium*, the five actions in this write event would be completed in one physical disk access, thus drastically reducing the I/O overhead.

synch level = low (default)
The operating system decides how and when to synchronize the data to disk. The impact of this option is more difficult to assess, because the rules are different for each operating system and file system.

The fault-tolerant switch manager - impact on performance

This section describes the impact that the enablement of the fault-tolerant switch manager feature has on the performance of the general architecture and the individual system. The fault-tolerant switch manager is enabled by setting the `enSwfaultTol` global option to *yes*. When it is set, the master domain manager distributes messages to all fault-tolerant agents with *FullStatus* set to *yes*. This option has not dynamic capabilities and is not designed to work with broker agents.

Enabling this option impacts the following:

- Network traffic
- Disk space

Note: The fault-tolerant switch manager facility is only available if all of the workstations in the domain are at version 8.2, fix pack level 4, or higher.

Network Traffic

Network traffic is unchanged under normal conditions, but is increased during the replay phase, according to your choice and only under special conditions.

The replay phase is an essential part of the processing performed by the **`switchmgr`** command. It occurs when the new domain manager processes its Symphony file against its copies of the messages received, as it attempts to update its copy of the Symphony file.

Under normal conditions, the outbound reliability does not create any additional network traffic, because the messages are only stored for an eventual replay operation. The multiple inbound connections do not generate additional traffic because the traffic that was previously copied by the domain manager to the *FullStatus* member is now copied to the *FullStatus* members directly by the fault-tolerant agents.

During the replay phase, the connection protocol initiated by mailman on the backup domain manager includes a new phase for the replay of messages not sent

by the failed domain manager. The impact of the message replay might be important, depending on the number of messages "trapped" in the old domain manager.

Disk Space

There are two places within the network where disk space use increases following the activation of the additional fault tolerance.

These places are as follows:

- On the single fault-tolerant agent. Here, in addition to the `tomaster.msg` queue, new queues are created for the other *FullStatus* fault-tolerant agents. These queues need not be considered, because the impact on a single agent is small.
- On the *FullStatus* fault-tolerant agents acting as backup domain managers. Here new ftbox message files are created. Upward traffic to the upper domain manager is in `ftbox/ftup.msg` and downward traffic to the lower domain manager is in `ftbox/ftdown.msg`.

Scalability

In an environment with large numbers of scheduling objects, the following impacts are felt:

- "Impact on **JnextPlan**"
- "Impact on reporting" on page 492
- "Impact on event rule deployment" on page 492

The resolution for these problems often includes making the following changes:

- "Increasing application server heap size" on page 492
- "Increasing maximum DB2 log capacity" on page 493

Impact on JnextPlan

The main impact on performance caused by a large network of workstations running many jobs over a production period of many days, is on **JnextPlan**. The key factor is the number of job stream instances that **JnextPlan** needs to handle. **JnextPlan** has to process each of these instances, and the time it takes to do so is a factor that can only be reduced by ensuring that the master domain manager and the database are on the most powerful computers possible, and that the communication, whether in local or remote, between the master domain manager and the database is maximized.

However, there are some specific measures that need to be taken as the number of jobs or job stream instances increases:

Number of jobs in the plan exceeds 40 000

In this event you need to increase the Java heap size used by the application server. The default is 512 MB, and you should at least double the heap size when job numbers exceed this level. Follow the procedure in "Increasing application server heap size" on page 492.

You have a large number of job stream instances in the plan

DB2 The default DB2 transaction log files cannot handle more than the transactions generated by about 180 000 job stream instances. You need to change the parameters that control the log file sizes or the numbers of log files that can be created, or both. Follow the procedure in "Increasing maximum DB2 log capacity" on page 493.

Oracle The number of transactions that can be managed by the Oracle log files depends on the way the Oracle database is configured. See the Oracle documentation for more details.

Note: If circumstances change and the number of job stream instances handled by **JnextPlan** falls below about 180 000, consider resetting the log and application server heap size settings to their default values, to avoid performance problems.

Impact on reporting

When a report is being processed, extra memory is required to handle large numbers of scheduling objects. The critical point is approximately 70 000 objects. This problem can be handled by increasing the Java heap size used by the application server. Follow the procedure in "Increasing application server heap size."

Impact on event rule deployment

When deploying large numbers of event rules, extra memory is required. The critical point is approximately 8 000 rules. This problem can be handled by increasing the Java heap size used by the application server. Follow the procedure in "Increasing application server heap size."

Increasing application server heap size

Follow this procedure to increase the Java heap size:

1. Log on to the computer where IBM Workload Scheduler is installed as the following user:

   Windows operating systems:
   Any user in the *Administrators* group.

   UNIX operating systems:
   root

2. Stop the WebSphere Application Server either by using the **conman stopappserver** command (see "Starting and stopping the application server and **appservman**" on page 450) or by running:

   Windows operating systems:
   `TWA_home\wastools\stopWas.bat`

   UNIX operating systems:
   `TWA_home/wastools/stopWas.sh`

3. Open the following file:

   `<WAS_profile_path>/config/cells/TWSNodeCell/nodes/TWSNode/servers/server1/server.xml`

   where the default value for *WAS_profile_path* is `<TWA_home>/WAS/TWSprofile`.

   Locate the following lines:

   ```
   <jvmEntries xmi:id="...."
   verboseModeClass="false"
   verboseModeGarbageCollection="false"
   verboseModeJNI="false"
   initialHeapSize="256"
   maximumHeapSize="1024"
   runHProf="false"
   ```

```
hprofArguments=""
debugMode="false"
debugArgs="...."/>
</jvmEntries>
```

4. Edit the `initialHeapSize` and `maximumHeapSize` fields to at least the values shown in Table 89.

Table 89. Heap size settings for **jvm 64 bits**

RAM (GB)	initialHeapSize (MB)	maximumHeapSize (MB)
2	512	1024
4	1024	2048
8	2048	4096

Note: Ensure that the computer RAM usage can handle whatever increased size you choose. If you have a **jvm 32 bits** on your machine, the maximum value for `maximumHeapSize` is *1536* MB.

5. Save the file `server.xml`
6. Start the WebSphere Application Server, either by using the **conman startappserver** command (see "Starting and stopping the application server and **appservman**" on page 450) or by running

   Windows operating systems:
   `<TWA_home>\wastools\startWas.bat`

   UNIX operating systems:
   `<TWA_home>/wastools/startWas.sh`

Increasing maximum DB2 log capacity

The IBM Workload Scheduler DB2 database uses a transaction log the maximum size of which is fundamentally important for the successful running of **JnextPlan** on very large databases.

The default log consists of 40 primary log files, which are always present, and 20 secondary log files, created on demand. Each file is about 4 MB in size, so the maximum log capacity using all of the "secondary" log files as well as the primary files is (40 + 20) x 4 MB = 240 MB.

The log space used by **JnextPlan** is dependent on the size of the preproduction plan. Approximately every 1000 job stream instances generate transactions that occupy 1 MB of space in the log file. Thus, the log files by default have a maximum theoretical capacity of 240 000 job stream instances. However, in practice, you should allow for at least 25% more space than this algorithm indicates, so the capacity of the default log files is around 180 000 job stream instances.

If **JnextPlan** has neared or exceeded that level, you must make more log space available to DB2.

In addition to performing the above calculation, you can also determine the log space actually used by a specific instance of **JnextPlan** and base your log size requirement on that figure.

Determining actual DB2 log file usage

The following is the procedure to verify how much space was used by a successful instance of the **JnextPlan** command:

1. After **JnextPlan** has run, log on to the computer where the IBM Workload Scheduler DB2 server is installed, as the DB2 instance owner (UNIX) or DB2 Administrator (Windows).
2. Open a DB2 command line window or shell, as follows:

   UNIX Follow these steps:

   a. Issue the command **su - db2inst1**, or change to the subdirectory `sqllib` of the home directory of the owner of the DB2 instance (by default *db2inst1*)

   b. Launch the command **. ./db2profile**

   Windows

   Select from the **Start** menu, **Programs** → **IBM DB2** → **Command Line Tools** → **Command Window**
3. Run the following command:

   ```
   db2 "get snapshot for database on TWS" > snapdb.txt
   ```

   where "TWS" must be changed to the actual database name if different
4. Open the `snapdb.txt` file and look for a section like this:

   ```
   Log space available to the database (Bytes)= 244315359
   Log space used by the database (Bytes)     = 484641
   Maximum secondary log space used (Bytes)   = 0
   Maximum total log space used (Bytes)       = 581636
   Secondary logs allocated currently         = 0
   ```

   The value shown in "Maximum total log space used" is the actual space used for the DB2 logs. This space should be allocated to DB2 using primary log files only: therefore, you should change the number of primary log files and their size as necessary to meet this requirement as a minimum.

   In addition, you are recommended to allocate a secondary log space to DB2. A good choice for the secondary log files is half the number allocated for the primary files.

The snapshot command described in step 3 can be run at any time to keep track of the current usage of the DB2 log space, without a noticeable impact on performance. All metrics shown are useful to monitor the current allocation of DB2 primary and secondary logs at any time, and to determine any required changes.

Procedure for changing the maximum DB2 log capacity

Do this as follows:

1. Log on to the computer where the IBM Workload Scheduler DB2 server is installed, as the DB2 instance owner (UNIX) or DB2 Administrator (Windows).
2. Open a DB2 command line window or shell, as follows:

   UNIX Follow these steps:

   a. Issue the command **su - db2inst1**, or change to the subdirectory `sqllib` of the home directory of the owner of the DB2 instance (by default *db2inst1*)

   b. Launch the command **. ./db2profile**

   Windows

   Select from the **Start** menu, **Programs** → **IBM DB2** → **Command Line Tools** → **Command Window**
3. Run the following commands:

   ```
   db2 update db cfg for <database_name> using LOGFILSIZ <log_file_size>
   db2 update db cfg for <database_name> using LOGPRIMARY <primary_log_files>
   db2 update db cfg for <database_name> using LOGSECOND <secondary_log_files>
   ```

where:

<*database_name*>
The name of the database:

- If you are running this from the computer where the DB2 server is installed, the installed default name is *TWS*. Supply this value unless you have changed it.
- You are not recommended to run this procedure from the computer where the DB2 client is installed, but if you do so, the installed default name is *TWS_DB*. Supply this value unless you have changed it.

<*log_file_size*>
The log file size in 4 KB pages. The default is 1000 (hence the default log file size of 4MB). Look in the DB2 documentation for details of the implications of choosing a larger or a smaller log file size. The maximum value is 262 144 (making the maximum log file size about 1 GB).

<*primary_log_files*>
The number of primary log files. The default is 40. The total maximum number of log files that DB2 can handle (primary and secondary) is 256. Thus, there is a maximum limit of 256 GB for the log, or approximately 256 million Job Scheduler instances! (maximum 256 files x 1 GB maximum file size)

<*secondary_log_files*>
The number of secondary log files. The default is 20. If there is enough free space on the file system, these additional log files are dynamically allocated by DB2 as needed (with a small impact on the performance of **JnextPlan**). Because these are only created if required, it is preferable to increase the number of secondary files, rather than the primary files. Typically, you allocate 50% of the primary log file value.

In making the calculation to allocate the log files, allow at least 25% more space than you think you require, to avoid that any slight miscalculation causes **JnextPlan** to fail.

Example: if you have determined from the procedure described in "Determining actual DB2 log file usage" on page 493 that **JnextPlan** has a current use of 320 MB, you could calculate as follows:

a. Increase 320 MB by 25%, giving 400 MB
b. Determine if you want more log files, or bigger log files, or both, by reference to the DB2 documentation. For example, you could choose to allocate 40 files with a size of 10 MB, 80 files with a size of 5 MB, or 100 files with a size of 4 MB. For the sake of this example, assume you have chosen 80 files with a size of 5 MB, so your LOGPRIMARY value will be 80.
c. Determine the log file size in 4 KB pages to give a log file size of 5 MB - your LOGFILSIZ value will thus be 1250.
d. Determine how many secondary log files are required. If you follow the 50% guideline you will need a LOGSECOND value of 40.

4. Log on to the computer where IBM Workload Scheduler is installed as the following user:

UNIX root

Windows
Any user in the *Administrators* group.

5. Access the directory: `TWA_home/wastools`
6. Stop the WebSphere Application Server using the **conman stopappserver** command (see "Starting and stopping the application server and **appservman**" on page 450)
7. On the computer where the DB2 server is installed, stop and start DB2, as follows:
   a. Ensure that no other applications are using this instance of DB2, or if they are that they can be stopped.
   b. Issue the following command:
      db2stop
   c. Issue the following command:
      db2start

   Note: It is strongly recommended that you stop and start DB2. If this is a problem for you, you must at least disconnect all applications from the DB2 instance and reconnect them. DB2 will apply the new parameters when you reconnect. If necessary, use the following command to force the disconnection of all open connections:

   ```
   db2 "force application all"
   ```
8. Start the WebSphere Application Server using the **conman startappserver** command (see "Starting and stopping the application server and **appservman**" on page 450)

Multiple Dynamic Workload Console production plan reports

From the Dynamic Workload Console you can launch production plan reports. These are heavy users of CPU time, and if they are requested for the entire plan, they can also take some considerable time to produce. If several are running at once, they can have a noticeable impact on the performance of the master domain manager.

If you notice a degradation of performance, you can determine if there are any reports running by checking for the report work files, as follows;

1. Navigate to the operating system's temporary directory
2. Look for files that have the following file name template:

   ```
   TWS-<sequential_number>-extr
   ```

   Each report currently in progress has one of these work files open. The files are removed when the report is completed.
3. Check the dates of these files, and consider only recent files (if a report fails during production at any time, its file remains in the temporary directory until the next reboot of the master domain manager or you run an operating system cleanup process that discards all files in the temporary directory).

There is no direct action to take, as you must wait until the report completes for the performance to recover.

However, if you note that large numbers of reports are being issued, it might indicate the following scenario:

1. A user issues a report request, expecting it to be available immediately

2. When the report does not appear immediately, the user things it has hung, closes and reopens the browser, and reissues the report. The closing of the browser does not stop the report production.
3. The user might repeat this action several times.

In this case, you can take action to remind the user that the production of large reports can be time-consuming, and that it always better to wait.

Dynamic Workload Console - adjusting session timeout settings

About this task

The value assigned to the session timeout settings defines after how many minutes a user is automatically logged out from the WebSphere Application Server. If you plan to perform long running operations, or to have many users connected concurrently to the Dynamic Workload Console, or expect to have low performance on the system where the Dynamic Workload Console is installed, you might want to increase these values.

Perform these steps to change the values assigned to the timeout settings:

1. Open the configuration file:

   ```
   <JazzSM_profile_dir>\config\cells\JazzSMNode01Cell\
             nodes\JazzSMNode01\servers\server1\server.xml
   ```

   where, the default value of *<JazzSM_profile_dir>* is:

   On Windows operating systems

   ```
   C:\Program Files\IBM\JazzSM\profile
   ```

   On UNIX operating systems

   ```
   /opt/IBM/JazzSM/profile
   ```

2. In the file, search for `invalidationTimeout` in the following tag:

   ```
   <tuningParams xmi:id="TuningParams_1188622510500"
                       usingMultiRowSchema="false"
                       maxInMemorySessionCount="1000"
                       allowOverflow="true"
                       scheduleInvalidation="false"
                       writeFrequency="TIME_BASED_WRITE"
                       writeInterval="10"
                       writeContents="ONLY_UPDATED_ATTRIBUTES"
                       invalidationTimeout="30">
   ```

   This is the parameter that sets the HTTP session timeout. By default `invalidationTimeout` is set to 30, which means that a user is logged out automatically after 30 minutes of inactivity.
3. Set `invalidationTimeout` to an appropriate value for your environment and for the activities you plan to perform.
4. Save the file.
5. Open the configuration file:

   ```
   <JazzSM_profile_dir>\config\cells\JazzSMNode01Cell\applications\isclite.ear\
                         deployments\isclite\deployment.xml
   ```

6. In the file, search for `invalidationTimeout` in the following tag:

   ```
   <tuningParams xmi:id="TuningParams_1188878529796"
                       usingMultiRowSchema="false"
                       maxInMemorySessionCount="1000"
                       allowOverflow="true"
                       scheduleInvalidation="false"
   ```

```
writeFrequency="TIME_BASED_WRITE"
writeInterval="10"
writeContents="ONLY_UPDATED_ATTRIBUTES"
invalidationTimeout="30">
```

By default, `invalidationTimeout` is set to 30, which means that a user is logged out automatically after 30 minutes of inactivity.

7. Set `invalidationTimeout` to an appropriate value for your environment and for the activities you plan to perform.
8. Save the file.
9. Open the configuration file:

   `<JazzSM_profile_dir>\config\cells\JazzSMNode01Cell\security.xml`
10. In the file, search for `timeout` in the following tag:

```
<authMechanisms xmi:type="security:LTPA"
            xmi:id="LTPA_1" OID="oid:1.3.18.0.2.30.2"
            authContextImplClass="com.ibm.ISecurityLocalObjectTokenBaseImpl
.WSSecurityContextLTPAImpl"
            authConfig="system.LTPA"
            simpleAuthConfig="system.LTPA"
            authValidationConfig="system.LTPA"
            timeout="120"
            keySetGroup="KeySetGroup_lab237165Node01_1">
```

    By default `timeout` is set to 120, which means that a user is logged out automatically after 120 minutes regardless of whether the user performed any actions on the WebSphere Application Server.
11. Set the timeout value in the following section of the file to an appropriate value for your environment and for the activities you plan to perform.
12. Save the file.
13. Restart the WebSphere Application Server.

Chapter 12. Availability

This chapter describes factors that might affect the availability of IBM Workload Scheduler on a workstation. It covers the following topics:

- "Resolving user ID account on Windows operating systems"
- "Using a temporary directory on UNIX"

Resolving user ID account on Windows operating systems

About this task

IBM Workload Scheduler needs to resolve the user ID account on Windows operating systems to verify the security information.

Windows users can be classified as domain users or local users. Domain users are defined in the domain controller, while local users are defined in the workstations of the network.

For a domain user, IBM Workload Scheduler requests the primary domain controller (or any domain controller for Windows 2000 or 2003 Active Directory), to identify an available domain controller. It then uses this domain controller identity to type out the structure for the user.

For a local user, IBM Workload Scheduler makes a request to the local workstation. Generally, IBM Workload Scheduler specifies two cases: one for the IBM Workload Scheduler user and one for the streamlogon user.

The following is a list of steps that IBM Workload Scheduler performs to authenticate Windows users, and the APIs involved:

1. IBM Workload Scheduler looks up the user in the reference domain. For the domain user, the reference domain is the name of the Windows network. For the local user, it is the name of the local workstation.

   API: `LookupAccountName`.
2. If the user is a domain user, IBM Workload Scheduler asks the primary domain controller for any domain controller that is available to resolve the account for the user in the reference domain.

   API: `NetGetAnyDCName` for Windows or `DsGetDcName` for Windows 2000 or 2003.
3. IBM Workload Scheduler requests the domain controller (or the local workstation if the user is local) for information about the user.

   API: `NetUserGetInfo`.

   Note: On Windows 2000 and 2003, the permissions for this API are contained in the `BUILTIN\"Pre-Windows 2000 compatible access"` group.

Using a temporary directory on UNIX

When performing IBM Workload Scheduler operations on UNIX, temporary files are written to the temporary directory on the local workstation. Ensure that the *<TWS_user>* running operations has *read* and *write* access to this directory.

Chapter 13. License Management in IBM License Metric Tool

According to your IBM Workload Scheduler license, IBM® License Metric Tool helps you maintain your license compliance. By using License Metric Tool, you can generate reports that summarize your license consumption. The generated reports are maintained on the License Metric Tool server and should be periodically reviewed and signed, creating a history for audit purposes in the process. If you are contacted by a third-party software compliance auditor who plans to visit your enterprise to carry out a software audit, ensure that all reports are up-to-date and signed, and then supply copies of reports that cover the time periods that the auditor requests.

The following IBM Workload Scheduler license models are available to customers:

- "Processor Value Unit license model"
- "Per Job license model" on page 504
- "Using per Job queries after upgrading to version 9.4, Fix Pack 2" on page 509

To install and configure IBM® License Metric Tool, see License Metric Tool V9.2.0 on the IBM Knowledge Center: http://www-01.ibm.com/support/knowledgecenter/SS8JFY_9.2.0/com.ibm.lmt.doc_9.2/com.ibm.license.mgmt.doc/ic-homepage_lmt.html.

Processor Value Unit license model

About this task

IBM® License Metric Tool generates reports to help you maintain compliance with your Processor Value Unit (PVU) sub-capacity license terms. License Metric Tool can calculate PVU consumption only when **software identification tags** exist and are activated by the customer. The **optman** global option **licenseType** must be set to **perServer** (default value) to activate **Processor Value Unit** consumption tracking. You can also set the **optman** global option **licenseType** to **byWorkstation** to specify that the licensing type (either **perServer** or **perJob**) is specified at creation time for each workstation.

License Metric Tool automatically detects the following IBM Workload Scheduler V9.4.0 chargeable components that are part of the product:

Table 90. Chargeable software components automatically detected by License Metric Tool

Chargeable Software Components
IBM Workload Scheduler agent V9.4.0
IBM Workload Scheduler agent for z/OS V9.4.0

To detect and count remote nodes that are managed by IBM Workload Scheduler chargeable components, you must manually deploy software tags because no product code is present on those nodes. License Metric Tool generates and assigns dedicated software tags during the creation of the Readiness Package for the latest release of IBM Workload Scheduler.

The following IBM Workload Scheduler Version 9.4.0 chargeable components require manual deployment of software tags:

Table 91. Chargeable software components that require software tag deployment on managed nodes

Chargeable Software Components	Assigned Software tags
IBM Workload Scheduler agent-less 9.4.0	ibm.com_IBM_Workload_Scheduler_agent-less-9.4.0.swidtag
IBM Workload Scheduler for third-party Applications 9.4.0	ibm.com_IBM_Workload_Scheduler_for_Third_Party_Applications-9.4.0.swidtag
IBM Workload Scheduler for IBM Applications 9.4.0	ibm.com_IBM_Workload_Scheduler_for_IBM_Applications-9.4.0.swidtag

Note: For information about how to match chargeable software components with the IBM Workload Scheduler access methods and application plug-ins, see table Table 92 on page 504

Complete the following procedure to manually deploy dedicated software tags **on each managed node** and calculate PVU consumption:

1. Identify the remote nodes that are managed by each of your chargeable components. For example, the remote nodes that are managed by the application server that you are connecting to, with IBM Workload Scheduler plug-in for IBM InfoSphere DataStage.
2. Extract assigned software tags from the `ILMT_IWS_for_Applications_and_agentless`.zip file that is included in your Agent installation media.
3. Place the assigned software tag anywhere on the managed node.
4. Wait for the next software scan in License Metric Tool to have the chargeable software components reported.
5. It is recommended that you check whether License Metric Tool reporting matches with currently managed nodes before signing each License Metric Tool report.

The master domain manager centrally maintains the history of the plug-in jobs that you run in your environment. The history can be used:

- During audits to verify which systems are actually managed by IBM Workload Scheduler.
- To check periodically the result of your License Metric Tool reporting.
- To verify your PVU license entitlement.

An SQL query is provided to access the history in the database. You can run the query either from the command-line interface of your database or by creating your custom SQL report tasks from the Dynamic Workload Console as described in *Dynamic Workload Console User's Guide*.

For each job definition in the database, the SQL query returns the:

1. Type of the plug-in job.
2. Name of the workstation on which the job is defined.
3. Name of the job.
4. XML file containing the name of the remote server on which the job ran. If the XML file does not contain the name of the remote server, you can find it in the plug-in properties file in the `TWS/JavaExt/cfg` agent folder.

- For DB2, IDS, MSSQL database types:

```
SELECT JOD_TASK_TYPE as Job_type, WKC_NAME as Workstation_name, JOD_NAME as Job_name,
JOD_TASK_STRING as Job_definition
FROM MDL.JOD_JOB_DEFINITIONS J inner join MDL.WKC_WORKSTATION_CLASSES W on J.WKC_ID=W.WKC_ID
WHERE JOD_BY_JSDL='Y' and UPPER(JOD_TASK_TYPE) NOT IN ('EXECUTABLE', 'DISTRIBUTEDSHADOWJOB',
'ZSHADOWJOB')
ORDER BY JOD_TASK_TYPE, WKC_NAME, JOD_NAME
```

- For Oracle database type:

```
SELECT JOD_TASK_TYPE as Job_type, WKC_NAME as Workstation_name, JOD_NAME as Job_name,
JOD_TASK_STRING as Job_definition
FROM twsuser.JOD_JOB_DEFINITIONS J inner join twsuser.WKC_WORKSTATION_CLASSES W
on J.WKC_ID=W.WKC_ID
WHERE JOD_BY_JSDL='Y' and UPPER(JOD_TASK_TYPE) NOT IN ('EXECUTABLE', 'DISTRIBUTEDSHADOWJOB',
'ZSHADOWJOB')
ORDER BY JOD_TASK_TYPE, WKC_NAME, JOD_NAME
```

where **twsuser** is the name of the IBM Workload Scheduler schema.

The following example shows you the query output for a **Datastage** job type:

```
datastage NY_1 DS_JOB <?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
xmlns:jsdldatastage="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdldatastage"
name="DATASTAGE">
  <jsdl:application name="datastage">
    <jsdldatastage:datastage>
   <jsdldatastage:DataStageParameters>
    <jsdldatastage:DataStagePanel>
     <jsdldatastage:Logon>
      <jsdldatastage:Domain>ncxx4175.romelab.it.ibm.com:9080</jsdldatastage:Domain>
      <jsdldatastage:Server>ncxx4175</jsdldatastage:Server>
      <jsdldatastage:UserName>isadmin</jsdldatastage:UserName>
      <jsdldatastage:password>{aes}ScWNLDAHuN9X5sbtvAVky3RVd7gOkJqNerDbFbpwrDg=
</jsdldatastage:password>
     </jsdldatastage:Logon>
     <jsdldatastage:JobDefinitionGroup>
      <jsdldatastage:ProjectNameGroup>
       <jsdldatastage:ProjectName>tws4apps</jsdldatastage:ProjectName>
      </jsdldatastage:ProjectNameGroup>
      <jsdldatastage:JobNameButtonGroup>
       <jsdldatastage:JobNameRadioButton>
        <jsdldatastage:JobName>dsj01_succ</jsdldatastage:JobName>
       </jsdldatastage:JobNameRadioButton>
      </jsdldatastage:JobNameButtonGroup>
      <jsdldatastage:FileRemotePath/>
     </jsdldatastage:JobDefinitionGroup>
     <jsdldatastage:JobExecutionGroup/>
    </jsdldatastage:DataStagePanel>
    <jsdldatastage:OptionsPanel>
     <jsdldatastage:JobOptionsGroup>
      <jsdldatastage:WarningLimitButtonGroup>
       <jsdldatastage:NoWarningLimitButton/>
      </jsdldatastage:WarningLimitButtonGroup>
      <jsdldatastage:RowLimitButtonGroup>
       <jsdldatastage:NoRowLimitButton/>
      </jsdldatastage:RowLimitButtonGroup>
      <jsdldatastage:OperationalMetadataGroup>
       <jsdldatastage:UseDefault/>
      </jsdldatastage:OperationalMetadataGroup>
     </jsdldatastage:JobOptionsGroup>
    </jsdldatastage:OptionsPanel>
   </jsdldatastage:DataStageParameters>
  </jsdldatastage:datastage>
  </jsdl:application>
</jsdl:jobDefinition>
```

Table 92. IBM Workload Scheduler chargeable access methods and application plug-ins

IBM Workload Scheduler access methods and application plug-ins	**Chargeable Software Components**
Remote Command,	IBM Workload Scheduler agent-less V9.4.0
Unixssh	
SAP R/3	IBM Workload Scheduler for third-party Applications V9.4.0
SAP PI Channel	
SAP BusinessObjects BI	
PeopleSoft	
Oracle E-Business Suite	
Informatica PowerCenter	
Salesforce	
Hadoop Map Reduce	
Hadoop Distributed File System	
Apache Oozie	
Apache Spark	
Amazon EC2	
Microsoft Azure	
IBM Sterling Connect:Direct	IBM Workload Scheduler for IBM Applications V9.4.0
IBM WebSphere MQ	
IBM InfoSphere DataStage	
IBM Cognos	
IBM BigInsights	
IBM Cloudant	
IBM SoftLayer	
z/OS	Paid by PVU, manually counted, ILMT not available

Per Job license model

About this task

To generate a report that summarizes your monthly per-job license usage, you can generate a license metric tag file (SLMTag).

In the **optman** global options, use the **licenseType** keyword to define the pricing model. If you set the **licenseType** keyword to **byWorkstation**, you can then define for each single workstation the pricing model to be applied at the workstation creation time, either **perServer** or **perJob**. If you select the **perServer** setting in **optman**, see "Processor Value Unit license model" on page 501 for more information about tracking license consumption. The queries listed below apply when you select in **optman** either the **byWorkstation** or **perJob** pricing models and return the license consumption tracking. If you select the **byWorkstation** value, the queries listed below return the number of records with **license type=J** generated by successful jobs on workstations which are set to **license type=perJob**.

The master domain manager centrally maintains the history of the jobs that you run in your environment. By using the **optman** global option **statsHistory**, you can set the number of days for which you maintain the history of the jobs. To track your monthly per-job license usage, set the value of **statsHistory** to 400 (which is the default value).

For the SQL statement to generate the SLMTag file, see the following samples:

- For **DB2** database type:

```
SELECT '<SchemaVersion>2.1.1</SchemaVersion>
<SoftwareIdentity>
<Name>IBM Workload Scheduler</Name>
<PersistentId>3303c35cbc08435080502d621d5cdbff</PersistentId>
<InstanceId>/opt/IBM/TWA</InstanceId>
</SoftwareIdentity>' as xml from sysibm.sysdummy1
UNION
SELECT xml_metrics as xml
FROM (SELECT CONCAT ('<Metric logTime="',CONCAT(current_date,CONCAT('T',
CONCAT(replace(current_time, '.', ':'),CONCAT('+00:00">
<Type>MONTHLY_JOBS</Type>
<Value>',CONCAT(JobNbr,CONCAT('</Value>
<Period><StartTime>',CONCAT(Year,CONCAT('-',
CONCAT(trim(VARCHAR_FORMAT(Month,'00')),CONCAT('-01T00:00:01+00:00</StartTime>
<EndTime>',CONCAT(Year,CONCAT('-',CONCAT(trim(VARCHAR_FORMAT(Month,'00')),
CONCAT('-',CONCAT(LAST_DAY,'T23:59:00+00:00</EndTime></Period>
</Metric>'))))))))))))))  as xml_metrics
FROM (SELECT Year, Month,
CASE
when Month = 2 then '28'
when Month = 4 then '30'
when Month = 6 then '30'
when Month = 9 then '30'
when Month = 11 then '30'
else '31'
end as LAST_DAY,
COUNT(*) AS JobNbr,
current date as current_date,
current time as current_time
from (SELECT unique year(Job_run_date_time) AS Year,
month(Job_run_date_time)AS Month, day(Job_run_date_time) AS day,
JOB_STREAM_WKS_NAME_IN_RUN, JOB_STREAM_NAME_IN_RUN,
JOB_NAME_IN_RUN FROM MDL.JOB_HISTORY_V
WHERE Job_status='S' and Workstation_license_type='J' and
 Actual_workstation_name_in_run not in
(select WKS_NAME from MDL.WKS_WORKSTATIONS where WKS_AGENT_TYPE='E'))
GROUP BY Year, Month))
ORDER BY xml desc
```

- For **Oracle** database type:

```
SELECT '<SchemaVersion>2.1.1</SchemaVersion>
<SoftwareIdentity>
<Name>IBM Workload Scheduler</Name>
<PersistentId>3303c35cbc08435080502d621d5cdbff</PersistentId>
<InstanceId>/opt/IBM/TWA</InstanceId>
</SoftwareIdentity>' as xml from dual
UNION
SELECT xml_metrics as xml
FROM (SELECT '<Metric logTime="' || cdate || 'T' || ctime || '+00:00">
<Type>MONTHLY_JOBS</Type>
<Value>' || JobNbr || '</Value>
<Period>
<StartTime>' || Year || '-' || trim(TO_CHAR(Month,'00')) || '-01T00:00:01+00:00</StartTime>
<EndTime>' || Year || '-' || trim(TO_CHAR(Month,'00')) || '-' || LAST_DAY || 'T23:59:00+00:00
</EndTime></Period></Metric>'  as xml_metrics
FROM (SELECT Year, Month,
CASE
when Month = 2 then '28'
when Month = 4 then '30'
when Month = 6 then '30'
when Month = 9 then '30'
when Month = 11 then '30'
else '31'
end as LAST_DAY,
CAST(COUNT(*) AS INT) AS JobNbr,
TO_CHAR(SYSDATE, 'YYYY-MM-DD') as cdate,
TO_CHAR(SYSDATE, 'HH24:MI:SS') as ctime
from (
SELECT unique EXTRACT(year FROM Job_run_date_time) AS Year,
  EXTRACT(month FROM Job_run_date_time) AS Month,
  EXTRACT(day FROM Job_run_date_time) AS Day,
  JOB_STREAM_WKS_NAME_IN_RUN,
  JOB_STREAM_NAME_IN_RUN,
  JOB_NAME_IN_RUN
  FROM JOB_HISTORY_V
  WHERE Job_status='S' and Workstation_license_type='J' and
   Actual_workstation_name_in_run not in
  (select WKS_NAME from WKS_WORKSTATIONS where WKS_AGENT_TYPE='E'))
GROUP BY Year, Month))
ORDER BY xml desc
```

- For **IDS** database type:

```
SELECT '<SchemaVersion>2.1.1</SchemaVersion><SoftwareIdentity>
  <Name>IBM Workload Scheduler</Name>
  <PersistentId>3303c35cbc08435080502d621d5cdbff</PersistentId><InstanceId>/
  opt/IBM/TWA</InstanceId></SoftwareIdentity>' as xml FROM SYSTABLES
UNION
SELECT xml_metrics as xml FROM (
SELECT CONCAT ('<Metric logTime="',CONCAT(current_date,CONCAT('T',CONCAT
 (current_time,CONCAT('+00:00"><Type>MONTHLY_JOBS</Type><Value>',CONCAT
 (round(JobNbr,0),CONCAT('</Value><Period><StartTime>',CONCAT(Year,CONCAT
 ('-',CONCAT(Month,CONCAT('-01T00:00:01+00:00</StartTime><EndTime>',CONCAT
 (Year,CONCAT('-',CONCAT(Month,CONCAT('-',CONCAT(LAST_DAY,'T23:59:00+00:00</EndTime>
 </Period></Metric>'))))))))))))))  as xml_metrics
FROM (SELECT Year,
replace(TO_CHAR(Month, "**"),'*','0') AS Month,
CASE
when Month = 2 then '28'
when Month = 4 then '30'
when Month = 6 then '30'
when Month = 9 then '30'
when Month = 11 then '30'
else '31'
end as LAST_DAY,
COUNT(*) AS JobNbr,
TO_CHAR(today,'%Y-%m-%d') as current_date,
TO_CHAR(extend (current, hour to second),'%H:%M:%S') as current_time
FROM (SELECT unique year(Job_run_date_time) AS Year,
month(Job_run_date_time) AS Month, day(Job_run_date_time) AS day,
JOB_STREAM_WKS_NAME_IN_RUN, JOB_STREAM_NAME_IN_RUN, JOB_NAME_IN_RUN
FROM MDL.JOB_HISTORY_V
WHERE Job_status='S' and Workstation_license_type='J' and
 Actual_workstation_name_in_run not in
(select WKS_NAME from MDL.WKS_WORKSTATIONS where WKS_AGENT_TYPE='E'))
GROUP BY Year, Month))
ORDER BY xml desc
```

- For **MSSQL** database type:

```
SELECT '<SchemaVersion>2.1.1</SchemaVersion>
<SoftwareIdentity>
<Name>IBM Workload Scheduler</Name>
<PersistentId>3303c35cbc08435080502d621d5cdbff</PersistentId>
<InstanceId>/opt/IBM/TWA</InstanceId>
</SoftwareIdentity>' as xml
UNION
SELECT b.xml_metrics as xml
FROM (SELECT '<Metric logTime="' + CONVERT(nvarchar(19), a.datetime, 126) + '+00:00">
<Type>MONTHLY_JOBS</Type>
<Value>' + CONVERT(varchar(10), a.JobNbr) + '</Value><Period>
<StartTime>' + CONVERT(varchar(10), a.Year) + '-' + RIGHT('00' + CONVERT(varchar(2),
a.Month), 2) + '-01T00:00:01+00:00</StartTime>
<EndTime>' + CONVERT(varchar(10), a.Year) + '-' + RIGHT('00' + CONVERT(varchar(2),
a.Month), 2) + '-' + a.LAST_DAY + 'T23:59:00+00:00</EndTime>
</Period></Metric>'as xml_metrics
FROM (SELECT Year, Month,
CASE
when Month = 2 then '28'
when Month = 4 then '30'
when Month = 6 then '30'
when Month = 9 then '30'
when Month = 11 then '30'
else '31'
end as LAST_DAY,
COUNT(*) AS JobNbr, SYSDATETIME() as datetime,
CONVERT (date, SYSDATETIME()) as cdate,
CONVERT (time, SYSDATETIME()) as ctime
FROM (SELECT distinct year(Job_run_date_time) AS Year,
month(Job_run_date_time) AS Month, day(Job_run_date_time) AS day,
Job_stream_wks_name_in_run, Job_stream_name_in_run, Job_name_in_run
FROM MDL.JOB_HISTORY_V
WHERE Job_status='S' and Workstation_license_type='J' and
 Actual_workstation_name_in_run not in
(select WKS_NAME from MDL.WKS_WORKSTATIONS where WKS_AGENT_TYPE='E')) r
GROUP BY Year, Month) a) b
ORDER BY xml desc
```

The following example shows a license metric tag file with the monthly number of jobs that ran in your environment:

```
<SchemaVersion>2.1.1</SchemaVersion>
<SoftwareIdentity>
<Name>IBM Workload Scheduler</Name>
<PersistentId>3303c35cbc08435080502d621d5cdbff</PersistentId>
<InstanceId>/opt/IBM/TWA</InstanceId>
</SoftwareIdentity>
<Metric logTime="2015-05-13T08:50:43+00:00">
<Type>MONTHLY_JOBS</Type>
<Value>2</Value>
<Period><StartTime>2015-05-01T00:00:01+00:00</StartTime>
<EndTime>2015-05-31T23:59:00+00:00</EndTime></Period>
</Metric>
<Metric logTime="2015-05-13T08:50:43+00:00">
<Type>MONTHLY_JOBS</Type>
<Value>22</Value>
<Period><StartTime>2015-04-01T00:00:01+00:00</StartTime>
<EndTime>2015-04-30T23:59:00+00:00</EndTime></Period>
</Metric>
```

An SQL query is provided to access the job history in the database, to verify the number of jobs that you run every month in your environment.

You can run the SQL query either from the command-line interface of your database, or by creating your custom SQL report tasks from the Dynamic Workload Console, as described in the related section in *Dynamic Workload Console User's Guide*.

- For **DB2** database type:

```
SELECT Year, Month, count(*) AS JobNbr from
(SELECT unique year(Job_run_date_time) AS Year,
month(Job_run_date_time) AS Month, day(Job_run_date_time) AS day,
JOB_STREAM_WKS_NAME_IN_RUN, JOB_STREAM_NAME_IN_RUN, JOB_NAME_IN_RUN
FROM MDL.JOB_HISTORY_V
WHERE Job_status='S' and Workstation_license_type='J' and
 Actual_workstation_name_in_run not in
(select WKS_NAME from MDL.WKS_WORKSTATIONS where WKS_AGENT_TYPE='E'))
GROUP BY Year, Month
```

- For **ORACLE** database type:

```
SELECT Year, Month, cast (count(*) AS INT) AS JobNbr from
(SELECT unique EXTRACT(year FROM Job_run_date_time) AS Year,
  EXTRACT(month FROM Job_run_date_time) AS Month,
  EXTRACT(day FROM Job_run_date_time) AS Day,
  JOB_STREAM_WKS_NAME_IN_RUN,
  JOB_STREAM_NAME_IN_RUN,
  JOB_NAME_IN_RUN
  FROM JOB_HISTORY_V
  WHERE Job_status='S' and Workstation_license_type='J' and
 Actual_workstation_name_in_run not in
  (select WKS_NAME from WKS_WORKSTATIONS where WKS_AGENT_TYPE='E'))
GROUP BY Year, Month
```

- For **IDS** database type:

```
SELECT Year, Month, count(*) AS JobNbr from
(SELECT unique year(Job_run_date_time) AS Year, month(Job_run_date_time) AS Month,
day(Job_run_date_time) AS day, JOB_STREAM_WKS_NAME_IN_RUN,
JOB_STREAM_NAME_IN_RUN, JOB_NAME_IN_RUN
FROM MDL.JOB_HISTORY_V
WHERE Job_status='S' and Workstation_license_type='J' and
 Actual_workstation_name_in_run not in
(select WKS_NAME from MDL.WKS_WORKSTATIONS where WKS_AGENT_TYPE='E'))
GROUP BY Year, Month
```

- For **MSSQL** database type:

```
SELECT Year, Month, count(*) AS JobNbr from
(SELECT distinct year(Job_run_date_time) AS Year,
month(Job_run_date_time) AS Month, day(Job_run_date_time) AS day,
Job_stream_wks_name_in_run, Job_stream_name_in_run, Job_name_in_run
FROM MDL.JOB_HISTORY_V
WHERE Job_status='S' and Workstation_license_type='J' and
 Actual_workstation_name_in_run not in
(select WKS_NAME from MDL.WKS_WORKSTATIONS where WKS_AGENT_TYPE='E')) r
GROUP BY Year, Month
```

Note: Entitlements are not required for repeated runs of the same Job in any Calendar Day. Calendar Day is defined as starting at 00:00 Greenwich Mean Time (GMT) and ending at 23:59 GMT.

Note: The SQL queries select only jobs that run successfully. The SQL queries do not count shadow jobs, jobs that run on agent for z/OS, and rerun jobs.

Using per Job queries after upgrading to version 9.4, Fix Pack 2

About this task

After you upgrade to version 9.4, Fix Pack 2, follow the steps listed below before you run the queries listed in "Per Job license model" on page 504:

1. Upgrade all components in your environment to version 9.4, Fix Pack 2.

2. Depending on your database, run one of the statements listed below as many times as necessary until the **History** table is entirely updated. Several runs might be necessary.
3. Run the queries listed in "Per Job license model" on page 504.

If you are using a DB2 database, run the following statement:

```
UPDATE ( SELECT * FROM MDL.JHR_JOB_HISTORY_RUNS
WHERE WKC_LICENSE_TYPE = '-' FETCH FIRST 10000 ROWS ONLY )
SET WKC_LICENSE_TYPE = 'J'
```

If you are using an Oracle database, run the following statement:

```
UPDATE JHR_JOB_HISTORY_RUNS
   set WKC_LICENSE_TYPE = 'J'
   where WKC_LICENSE_TYPE = '-' and rownum <= 10000
```

If you are using an IDS database, run the following statement:

```
UPDATE MDL.JHR_JOB_HISTORY_RUNS SET WKC_LICENSE_TYPE = 'J' WHERE JOB_ID  IN
 (SELECT JOB_ID FROM
    (SELECT FIRST 10000 JOB_ID FROM MDL.JHR_JOB_HISTORY_RUNS WHERE
    WKC_LICENSE_TYPE = '-')
  )
```

If you are using an MSSQL database, run the following statement:

```
UPDATE TOP (10000) MDL.JHR_JOB_HISTORY_RUNS set WKC_LICENSE_TYPE = 'J'
  where WKC_LICENSE_TYPE = '-'
```

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

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

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

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

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

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

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

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those

websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

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

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 Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

The performance data discussed herein is presented as derived under specific operating conditions. Actual results may vary.

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

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

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

© (your company name) (year).
Portions of this code are derived from IBM Corp. Sample Programs.
© Copyright IBM Corp. _enter the year or years_.

Trademarks

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

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

IT Infrastructure Library is a Registered Trade Mark of AXELOS Limited.

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

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

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

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

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

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

ITIL is a Registered Trade Mark of AXELOS Limited.

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

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the IBM website.

Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

Index

Special characters

Numerics

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

Q

R

S

T

U

V

W

X

Z

Product Number: 5698-WSH

Printed in USA