Tivoli® Workload Scheduler
Version 8.5.1 (Revised May 2010)

Troubleshooting Guide

IBM

SC32-1275-10

Tivoli® Workload Scheduler
Version 8.5.1 (Revised May 2010)

Troubleshooting Guide

SC32-1275-10

Note
Before using this information and the product it supports, read the information in "Notices" on page 171.

This edition applies to version 8, release 5, modification level 1, Fix Pack 1 of IBM Tivoli Workload Scheduler (program number 5698-WSH) and to all subsequent releases and modifications until otherwise indicated in new editions.

This edition replaces SC32-1275-09.

Contents

List of figures

List of tables

About this publication

IBM® Tivoli® Workload Scheduler: Troubleshooting provides information about troubleshooting IBM Tivoli Workload Scheduler and the Tivoli Dynamic Workload Console.

What is new in this release

For information about the new or changed functions in this release, see *Tivoli Workload Automation: Overview*.

For information about the APARs that this release addresses, see the Tivoli Workload Scheduler Download Document at http://www.ibm.com/support/docview.wss?rs=672&uid=swg24024804, and Tivoli Dynamic Workload Console Download Documents at http://www.ibm.com/support/docview.wss?rs=672&uid=swg24024805.

What is new in this release for troubleshooting

This section describes what has changed in this release with regard to troubleshooting since version 8.5.

What is new in this publication for version 8.5.1

This section describes what has changed in this publication since version 8.5.

Changed or added text with respect to the previous version is marked by a vertical bar in the left margin. If the changed or added text applies only after a particular fix pack is installed, the mark in the left margin is the number of the fix pack. For example, 1 to refer to Fix Pack 1, and so on.

- The tws_inst_pull_info script is a new tool that replaces the Metronome tool. You can use it to produce information about your Tivoli Workload Scheduler environment and take a snapshot of the database and configuration data on the master domain manager.
- Chapter 6, “Troubleshooting dynamic workload broker” is new
- A section about Using IBM Support Assistant is new
- Specific troubleshooting scenarios have been added or updated for:
  - Agent with dynamic scheduling capabilities cannot be found from Tivoli Dynamic Workload Console
  - Submitted job is not running on agent with dynamic scheduling capabilities
  - Job status of a submitted job is continually shown as running on agent with dynamic scheduling capabilities
  - Engine connection does not work when connecting to the z/OS connector (versions 8.3.x and 8.5.x)
  - Engine connection does not work when connecting to the z/OS connector V8.3.x or a distributed Tivoli Workload Scheduler engine V8.3.x
  - WebSphere does not start when using an LDAP configuration
  - Access Error received when launching a task from the browser bookmark
  - Language-specific characters are not correctly displayed in graphical views

- Plan View panel seems to freeze with Internet Explorer version 7
- Plan View limit: maximum five users from the same engine

What is new in this publication for version 8.5.1.1

This section describes what has changed in this publication since version 8.5.1.

- "Configuring traces for the Tivoli Workload Scheduler agent" on page 13 explains how to configure Tivoli Workload Scheduler traces without having to restart the agent.
- "DB2 might lock while making schedule changes" on page 88 documents this problem and provides solution for mitigating.
- "Log files grow abnormally large in mixed environment with version 8.4 or higher master domain manager and 8.3 or lower agents" on page 107 documents this problem and provides a workaround.
- "Plan View limit: maximum five users using the same engine" on page 141 documents this problem and provides a solution.

Changed or added text is marked with a 1 (one) on the left page margin.

Who should read this publication

This publication is designed to help users deal with any error situations they encounter while working with Tivoli Workload Scheduler. The publication includes targeted troubleshooting information about some specific activities and solutions to problems that you might encounter while running the product.

Some of these solutions need an expert user of Tivoli Workload Scheduler to resolve them, while others require the expertise of an expert systems programmer, who has a reasonable understanding of the Tivoli Workload Scheduler infrastructure and its inter-component interactions.

Publications

Full details of IBM Tivoli Workload Automation publications can be found in *Tivoli Workload Automation: Publications*, . This document also contains information on the conventions used in the publications.

A glossary of terms used in the product can be found in *Tivoli Workload Automation: Glossary*, .

Both of these are in the Information Center as separate publications.

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 with respect to the Tivoli Dynamic Workload Console, see the Accessibility Appendix in the *Tivoli Workload Scheduler: User's Guide and Reference*, SC32-1274.

For full information with respect to the Job Scheduling Console, see the Accessibility Appendix in the *Tivoli Workload Scheduler: Job Scheduling Console User's Guide*, SC32-1257.

Tivoli technical training

For Tivoli technical training information, see the following IBM Tivoli Education Web site:

http://www.ibm.com/software/tivoli/education

Support information

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:

Online
Go to the IBM Software Support site at http://www.ibm.com/software/support/probsub.html and follow the instructions.

IBM Support Assistant
The IBM Support Assistant (ISA) is a free local software serviceability workbench that helps you resolve questions and problems with IBM software products. The ISA provides quick access to support-related information and serviceability tools for problem determination. To install the ISA software, go to http://www.ibm.com/software/support/isa.

Troubleshooting Guide
For more information about resolving problems, see the problem determination information for this product.

For more information about these three ways of resolving problems, see Appendix A, "Support information," on page 161.

Chapter 1. Getting started with troubleshooting

This publication gives troubleshooting information about the Tivoli Workload Scheduler *engine*. The engine comprises the components of Tivoli Workload Scheduler that perform the workload scheduling activities, together with the command line by which they can be controlled.

Troubleshooting for other Tivoli Workload Scheduler activities, products, and components can be found in their relevant publications, as follows:

Table 1. Where to find other troubleshooting material

Activity, Product, or Component	**Publication**
Installation, upgrade, and uninstallation of Tivoli Workload Scheduler components and the Tivoli Dynamic Workload Console	*Tivoli Workload Scheduler: Planning and Installation Guide*, SC32-1273
Limited fault-tolerant agents for i5/OS®	*Tivoli Workload Scheduler: Limited Fault-tolerant Agent for i5/OS*, SC32-1280
Tivoli Workload Scheduler for z/OS®	*Tivoli Workload Scheduler for z/OS: Diagnosis Guide and Reference*, SC32-1261 *Tivoli Workload Scheduler for z/OS: Messages and Codes*, SC32-1267
Tivoli Workload Scheduler for Applications	*Tivoli Workload Scheduler for Applications: User's Guide*, SC32-1278
Tivoli Workload Scheduler for Virtualized Data Centers	*Tivoli Workload Scheduler for Virtualized Data Centers: User's Guide*, SC32-1454
Job Scheduling Console (V8.4)	*Tivoli Workload Scheduler: Job Scheduling Console User's Guide*, SC32-1257

Many of the procedures described in this publication require you to identify a file in the installation path of the product and its components. However, they can have more than one installation path, as described in "Where products and components are installed."

Where products and components are installed

This section describes how to determine where Tivoli Workload Scheduler components are installed, as follows:

Tivoli Workload Scheduler installation path

You can install more than one Tivoli Workload Scheduler component on a system. Each master domain manager, backup master domain manager, domain manager, backup domain manger, agent, or connector is installed in a path called a *Tivoli Workload Automation instance*. Only one component can be installed in each instance. The default path for the *Tivoli Workload Automation instance* is:

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*

The installation path of Tivoli Workload Scheduler is:

TWA_home/TWS

Dynamic workload broker installation path
The files pertinent to dynamic workload broker are installed in the following path:

TWA_home/TDWB

Tivoli Dynamic Workload Console installation path
The Tivoli Dynamic Workload Console can be installed in more than one path:

- It can be installed alongside Tivoli Workload Scheduler or alone in a *Tivoli Workload Automation instance* using the embedded version of WebSphere® Application Server. In this case its path is:

  TWA_home/TDWC
- It can be installed on your own external instance of WebSphere Application Server. In this case its path depends on where your instance of WebSphere Application Server is installed (except for the uninstaller, which is installed in a path of your choice). The administrative procedures in this publication do not address problems that occur with the external version of WebSphere Application Server.

  If you are using the Tivoli Dynamic Workload Console on an external version of WebSphere Application Server, and an administrative procedure refers to the path *TWA_home*/TDWC, substitute it with the installation path of the Tivoli Dynamic Workload Console on your external version of WebSphere Application Server

The embedded WebSphere Application Server installation path
The embedded WebSphere Application Server is automatically installed when you create a new *Tivoli Workload Automation instance*. Its installation path is:

TWA_home/eWAS

The command line client installation path
The command line client is installed outside all *Tivoli 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 embedded WebSphere Application Server is not supplied with an administration GUI, many of its administration tasks are performed by running tools supplied with Tivoli 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 Tivoli Workload Automation instances, see "Finding out what has been installed in which Tivoli Workload Automation instances"

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

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

1. Access the following directory:

   UNIX `/etc/TWA`

   Windows
   `%windir%\TWA`

2. List the contents of the directory. Each Tivoli 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 Tivoli Workload Scheduler environment.

   The contents are similar to this:

```
#TWAInstance registry
#Mon Nov 24 15:35:02 CET 2008
TWS_version=8.5.0.00
EWas_basePath=C\:/Program Files/IBM/TWA/eWAS
TWS_counter=1
EWas_counter=2
TWA_path=C\:/Program Files/IBM/TWA
TWS_server_name=twaserver
TDWC_version=8.5.0.0
TWS_instance_type=MDM
EWas_profile_path=C\:/Program Files/IBM/TWA/eWAS/profiles/twaprofile
EWas_node_name=DefaultNode
TWS_basePath=C\:\\Program Files\\IBM\\TWA\\TWS
EWas_user=twsuser85
EWas_cell_name=DefaultNode
TDWC_EXTERNAL_WAS_KEY=false
EWas_version=6.1.0.19
TDWC_counter=1
EWas_server_name=twaserver
EWas_update_installer_dir=C\:/Program Files/IBM/WebSphere/UpdateInstaller
TDWC_basePath=C\:/Program Files/IBM/TWA/TDWC
TWS_user_name=twsuser85
TWS_FIX_LIST_KEY=
TDWC_FIX_LIST_KEY=
TWA_componentList=TWS,EWas,TDWC
EWas_isc_version_key=7.1.0.06
EWas_profile_name=twaprofile
EWas_service_name=twsuser85
```

   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 Tivoli Workload Scheduler component is installed

TDWC Where the Tivoli Dynamic Workload Console is installed

eWAS Where the embedded WebSphere Application Server is installed

wastools
Where the tools that you use to configure embedded 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 Tivoli Workload Automation

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

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

MDM Master domain manager

BKM Backup master domain manager

FTA Agent or domain manager

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

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

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

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

The only component of Tivoli Workload Scheduler which is installed in a Tivoli 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 Tivoli 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 Tivoli 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.

Built-in troubleshooting features

Tivoli Workload Scheduler is supplied with the following features that assist you with troubleshooting:

- Informational messages that inform you of expected events.
- Error and warning messages that inform you of unexpected events.
- Message helps for the most commonly-occurring messages. See *Tivoli Workload Scheduler: Messages*.
- A logging facility that writes all types of messages to log files, which you use to monitor the progress of Tivoli Workload Scheduler activities. See "Generating engine log files with CCLog" on page 8.
- A Log Analyzer that you use to read, analyze and compare log files. See "Analyzing log files with Log Analyzer" on page 18.
- An auditing facility that provides an audit trail of changes to the Tivoli Workload Scheduler database and plan for use in both monitoring and troubleshooting. See "Maintaining an audit trail" on page 30.
- A configuration snapshot facility that provides IBM Software Support with First Failure Data Capture (FFDC) configuration information when unexpected events occur. See "Capturing trace information at failure time" on page 35.
- An autotrace facility that maintains a trace of the activities of all of the main components. It is operated by you, under the control of IBM Software Support, to provide them with very useful information after unexpected events have occurred. See "Keeping a trace record for the main components" on page 43 for more details.
- A facility that automatically creates an FFDC software autotrace snapshot if a component failure can be detected by its parent component. See "Autosnapshot (ffdc)" on page 52.

Keeping up to date with the latest fix packs

To avoid problems with Tivoli Workload Scheduler, install the latest fix packs. Fix packs contain fixes to problems that IBM, you, or other customers have identified. Install the latest fix pack when it becomes available, to avoid problems.

Upgrading your whole environment

To avoid problems with Tivoli Workload Scheduler, ensure that when you upgrade to a new version you do so across your whole environment.

The components of this version of Tivoli Workload Scheduler are compatible with components of many previous versions (see *Tivoli Workload Automation: Overview* for full details). However, running Tivoli Workload Scheduler in a mixed network increases the possibility of problems arising, if only because each new release of Tivoli Workload Scheduler not only adds functions, but also improves the stability and reliability of the various components. Try not to run in a mixed network for extended periods.

Chapter 2. Diagnostic tools

This chapter gives an overview of the Tivoli Workload Scheduler diagnostic tools. The following topics are described:

Understanding log and trace files

Tivoli Workload Scheduler and Tivoli Dynamic Workload Console maintain log files for different activities in different places. To find out about the log files for the Job Scheduling Console, see the *Tivoli Workload Scheduler: Job Scheduling Console User's Guide*.

This publication contains information about the Tivoli Workload Scheduler the log files of the main processes, sometimes called the "engine", and the Tivoli Dynamic Workload Console. It is divided into these sections:

Log and trace messages

Tivoli Workload Scheduler automatically merges logs and traces into one file. The resulting file is stored in the<*TWA_home*>/stdlist/trace directory with the following name: <*yyyymmdd*>_TWSMERGE.log, where yyyy = year, mm = month, and dd = day. Log messages are also logged in a separate file stored in the <*TWA_home*>/stdlist/log directory with the following name: <*yyyymmdd*>_TWSMERGE.log

Log messages

These are messages that provide you with information, give you warning of potential problems, and inform you of errors. Most log messages are described in *Tivoli Workload Scheduler: Messages*. Log messages are translated into the following languages:

- Chinese - simplified
- Chinese - traditional
- French
- German
- Italian
- Japanese
- Korean
- Portuguese - Brazilian

- Spanish

Messages are written to the log file in the language of the locale set on the computer where they were generated, at the moment when they were generated.

Trace messages
These are messages for IBM Software Support that provide in depth information about Tivoli Workload Scheduler processes. In most cases they are in English. Whereas log messages are written so that you can understand them in relation to the activity you were performing, trace messages might not be. There is no guarantee that you can diagnose any error situations from the information they contain.

You can use local option `restricted stdlists` to limit access to the `stdlist` directory on your UNIX workstation. See the *Tivoli Workload Scheduler: Administration Guide* for details.

Installation log files

For details of the installation log files, see *Tivoli Workload Scheduler: Planning and Installation Guide*.

Generating engine log files with CCLog

CCLog is a logging engine that creates log files in a defined structure. It can be used to monitor many products from a variety of software suppliers. The configuration supplied with Tivoli Workload Scheduler uses it uniquely for Tivoli Workload Scheduler's processes.

CCLog file locations

The log files it produces are stored in different places, depending on the settings in the `localopts` file:

merge stdlists = yes

- <*TWA_home*>/TWS/stdlist/logs/<*yyyymmdd*>_NETMAN.log
  This is the log file for netman.
- <*TWA_home*>/TWS/stdlist/logs/<*yyyymmdd*>_TWSMERGE.log
  This is the log file for all other processes

merge stdlists = no
<*TWA_home*>/TWS/stdlist/logs/<*yyyymmdd*>_<*process_name*>.log

where <*process_name*> is one of the following:

```
BATCHMAN
CONNECTR
JOBMAN
JOBMON
MAILMAN
NETMAN
WRITER
```

Low-level traces, and open source library messages that do not conform to the current standard Tivoli Workload Scheduler message format (for instance, some SSL stdout and stderror messages), are found in the following files: <*TWA_home*>/TWS/stdlist/yyyy.mm.dd/<*TWS_user*>. For more information, see the *Tivoli Workload Scheduler: User's Guide and Reference*.

CCLog switching

The Tivoli Workload Scheduler log files are switched every day, at the time set in the *startOfDay* global options (**optman**).

CCLog customization

You can customize the information written to the log files by modifying selected parameters in its properties file.

Note: Do not change any parameters in this file other than those detailed here, otherwise you might compromise the logging facility.

The CCLog properties file is as follows:

`<TWA_home>/TWS/TWSCCLog.properties`

where <*TWA_home*> is the directory where Tivoli Workload Scheduler is installed.

The parameters that can be modified are as follows:

fomatters.basicFmt.dateTimeFormat
This contains a specification of the date and time format used by CCLog when adding the date and time stamp to the message header. The format uses the standard *strftime* format convention, used by many programming libraries. The full format details can be found by searching the Internet, but a synthesis of the commonly used definitions is included in Appendix B, "Date and time format reference - strftime," on page 169.

fomatters.basicFmt.separator
This defaults to the pipe symbol "|", and is used to separate the header of each log message, which contains information such as the date and time stamp and the process that issued the error, from the body, which contains the process-specific information such as the issuing process, the message number and the message text. You can change the separator to another character or characters, or set it to null.

twsHnd.logFile.className
This indicates if CCLog uses semaphore memory to write to the log file. The default setting (`ccg_filehandler`) tells CCLog to write each line of a multiline message separately. Each process interleaves each line of its multiline messages with messages from other processes, if necessary, improving performance. While this approach could potentially make the log files more difficult to read, this interleaving only occurs in extreme situations of very high use, for example when many jobs are running concurrently.

The previous default setting (prior to version 8.2, fix pack 6) `ccg_multiproc_filehandler`, defines that each process completes writing any log message, including multiline messages, before freeing the log file for another process to use. This can have an impact on performance when many processes are running concurrently.

tws.loggers.className
This indicates the type of log layout you want to use, determining the number of fields in the log record header. The default setting (`ccg_basiclogger`) tells CCLog to put just the date/time stamp and the process name in the header. The alternative setting is `ccg_pdlogger`, which contains more information in the header, thus reducing the length of the log records available for the message text.

tws.loggers.msgLogger.level
Determines the type of messages that are logged. Change this value to log more or fewer messages, as appropriate, or on request from IBM Software Support. Valid values are:

INFO All log messages are displayed in the log. The default value.

WARNING
All messages except *informational* messages are displayed.

ERROR
Only *error* and *fatal* messages are displayed.

FATAL
Only messages which cause Tivoli Workload Scheduler to stop are displayed.

tws.loggers.organization
This defaults to *IBM* and is used to differentiate between log entries from applications from different suppliers when the same instance of CCLog is being used by more than one software supplier. Tivoli Workload Scheduler is supplied with a unique instance, and thus unique log files, so if this value is prefixed to your log messages, you can set the value of this parameter to null to avoid it being displayed.

tws.loggers.product
This defaults to *TWS* and is used to differentiate when the same log files are used by more than one product. Tivoli Workload Scheduler is supplied with unique log files, so if this value is prefixed to your log messages, you can set the value of this parameter to null to avoid it being displayed.

tws.loggers.trc<*component*>**.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
Only *error* and *critical* messages are written to the trace.

CRITICAL
Only messages which cause Tivoli Workload Scheduler to stop are written to the trace.

The component names used in the property names are for the most part self-explanatory, but the following short explanations might help:

Logger
The main internal component of Tivoli Workload Scheduler that performs the scheduling activities.

Sendevnt
The event processor.

Connectr
The connector.

Note: No other parameters must be modified.

CCLog performance

If you use the default configuration, CCLog does not normally have a significant impact on performance. If you believe that it is impacting performance, check that the default values for the parameters `twsHnd.logFile.className` and `twsloggers.className` are as described in "CCLog customization" on page 9, and have not been set to other values.

However, even if the default parameters are in use, you might find that in situations of very heavy workload, such as when you have many jobs running simultaneously on the same workstation, multiline log messages become interleaved with messages from other processes. The length of log messages has been increased to offset this risk, but if you find it becoming a problem, contact IBM Software Support for advice on how to reset the previous settings, which avoided the interleaved messages, but had an impact on performance at busy times.

Tivoli Dynamic Workload Console log and trace files

Table 2 lists the log files created by the Tivoli Dynamic Workload Console:

Table 2. Locations of log files and trace files

Path	Files	Content

If installed on the embedded WebSphere Application Server
`<TWA_home>/eWAS/profiles/twaprofile/servers/twaserver<n>/logs`
If installed on the external WebSphere Application Server:
`<tdwc_install_dir>/appserver/profiles/<your_profile>/servers/<your_server>/logs`
	`SystemOut.log`,
	`SystemErr.log`
	`trace.log`
		The Tivoli Dynamic Workload Console run time logs.

Table 2. Locations of log files and trace files (continued)

Path	Files	Content
On Windows: *%TEMP%*\TWA\tdwc85 **On UNIX:** *$TMPDIR*/TWA/tdwc85 if set, otherwise /tmp/TWA/tdwc85	tdwcinstall.log	The Tivoli Dynamic Workload Console installation log.
	tdwcuninstall.log	The Tivoli Dynamic Workload Console uninstall log.
	wsadmin.log	The trace file containing the information about the configuration procedures stored during the installation phase.
	securityConfig*nnnn*.log	The Tivoli Dynamic Workload Console log file containing the details about the installation errors reported in the tdwcinstall.log file. The numeric value *nnnn* is automatically assigned at installation time. Access the tdwcinstall.log file to read the filename of the securityConfig*nnnn*.log file.

Note: For information about the path represented by *tdwc_install_dir*, see the *Tivoli Workload Scheduler: Planning and Installation Guide*.

Activating traces in Tivoli Dynamic Workload Console

Run the following steps to activate the Tivoli Dynamic Workload Console traces at run time:

1. Log in to the Integrated Solutions Console as administrator.
2. In the navigation tree, click **Troubleshooting**, and then **Logs and Trace**.
3. Click the name of the server for which you want to run the traces, for example **tdwcserver**.
4. Click **Diagnostic Trace**.
5. Click **Change Log Detail Levels**.
6. Select:

   Configuration
   If you want to apply the changes to the trace settings after having restarted the server.

   Run time
   If you want to apply the changes to the trace settings without restarting the server.
7. Choose the packages for which you want to activate the traces. For the Tivoli Dynamic Workload Console traces, make this selection:
   a. Scroll down to **com.ibm.tws.***.
   b. Expand the tree.
   c. Click on **com.ibm.tws.webui.***.
   d. Click on **All Messages and Traces**.
   e. Click **OK**.
   f. Click **Save**.

8. Stop and start the server.

Alternatively, you can activate the Tivoli Dynamic Workload Console traces as follows:

1. Edit the following XML file:

   If installed on the embedded WebSphere Application Server:
   <*TWA_home*>/eWAS/profiles/twaprofile/config/cells/DefaultNode/nodes/DefaultNode/servers/twaserver<*n*>/server.xml

   If installed on the external WebSphere Application Server:
   <*tdwc_install_dir*>/appserver/profiles/<*your_profile*>/config/cells/<*your_cell*>/nodes/<*your_node*>/servers/<*your_server*>/server.xml
2. Change the value assigned to the property **startupTraceSpecification** from:

   com.ibm.tws.webui.*=info

   to:

   com.ibm.tws.webui.*=all.
3. Save the changes
4. Stop and start the server.

When you enable tracing at run time the traces are stored in the following file:

If installed on the embedded WebSphere Application Server:
<*TWA_home*>/eWAS/profiles/twaprofile/config/cells/DefaultNode/nodes/DefaultNode/servers/twaserver<*n*>/trace.log

If installed on the external WebSphere Application Server:
<*tdwc_install_dir*>/appserver/profiles/<*your_profile*>/config/cells/<*your_cell*>/nodes/<*your_node*>/servers/<*your_server*>/trace.log

Configuring traces for the Tivoli Workload Scheduler agent

Trace files are enabled by default for the Tivoli Workload Scheduler agent. To modify the related settings you can use one of the following options:

- Edit the [JobManager.Logging] section in the `JobManager.ini` file, as described in section *Configuring log and trace properties* in the *IBM Tivoli Workload Scheduler Administration Guide*. This procedure requires that you stop and restart the Tivoli Workload Scheduler agent.
- Use one or more of the following command-line commands, without stopping and restarting the Tivoli Workload Scheduler agent:
  - enableTrace
  - disableTrace
  - showTrace
  - changeTrace

The syntax for the commands is as follows:

enableTrace
Sets the trace to the maximum level, producing a verbose result.

disableTrace
Sets the traces to the lowest level.

showTrace [> *trace_file_name***.xml]**
Displays the current settings defined in the [JobManager.Logging] section of the `JobManager.ini` file for the Tivoli Workload Scheduler agent traces.

You can also redirect the [JobManager.Logging] section to a file to modify it. Save the modified file and use the **changeTrace** command to make the changes effective immediately.

changeTrace [*trace_file_name***.xml]**
Reads the file containing the modified trace settings and implements the changes immediately and permanently, without stopping and restarting the Tivoli Workload Scheduler agent.

On agents running UNIX and Linux, you can optionally run the `ita_props.sh` script to set the environment to *TWA_home*`/TWS/ITA/bin`, so that you can run these commands directly without having to specify the relative path.

The following file is an example of the file created by the **showTrace** command:

```
<?xml version="1.0" encoding="UTF-8"?><jmgr:updateConfigurationResponse
                      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                      xmlns:jmgr="http://www.ibm.com/xmls/prod/scheduling
/1.0/JobManager"><jmgr:Section name="JobManager.Logging.cclog">
                      <jmgr:Property>
                                      <jmgr:Name>JobManager.trfl.level
</jmgr:Name>
                                      <jmgr:Value>1011</jmgr:Value>
                                </jmgr:Property>
                                <jmgr:Property>
                                      <jmgr:Name>JobManager.trhd.maxFileBytes
</jmgr:Name>
                                      <jmgr:Value>1024000</jmgr:Value>
                                </jmgr:Property>
                                <jmgr:Property>
                                      <jmgr:Name>JobManager.trhd.maxFiles
</jmgr:Name>
                                      <jmgr:Value>4</jmgr:Value>
                           </jmgr:Property>
              </jmgr:updateConfigurationResponse>
```

where:

JobManager.trfl.level
Defines the quantity of information to be provided in the traces. The value ranges from 0 to 3000. Smaller numbers correspond to more detailed tracing. The default is 3000.

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

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

Log and trace files for the application server

The log and trace files for the application server can be found in:

The embedded WebSphere Application Server:
<*TWA_home*>/eWAS/profiles/twaprofile/servers/twaserver<*n*>/logs

The WebSphere Application Server:
<*tdwc_install_dir*>/appserver/profiles/<*your_profile*>/servers/<*your_server*>/logs

Setting the traces on the embedded WebSphere Application Server for the major Tivoli Workload Scheduler processes

The application server handles all communications between the Tivoli Workload Scheduler processes. The trace for these communications is set to "tws_info" by default (information messages only). The application server can be set to trace "all" communications, either for the whole product or for these specific groups of processes:

- Command line
- Connector
- Database
- Planner
- Utilities
- Dynamic workload broker

Significant impact on performance: Activating traces for the embedded WebSphere Application Server leads to a significant impact on performance, especially if you set the tracing to "all". Thus you are strongly advised to identify the process group where the problem that you want to trace is occurring, and only set the trace to that group.

The procedure for changing the trace level on the embedded WebSphere Application Server is as follows:

1. Log on to the computer where Tivoli 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 and restart the application server, as described in the section on starting and stopping the application server in the *Tivoli Workload Scheduler: Administration Guide*.
4. Run the script:

   UNIX

   ```
   ./changeTraceProperties.sh -user <TWS_user>
                              -password <TWS_user_password>
                              -mode <trace_mode>
   ```

   Windows

   ```
   changeTraceProperties.bat -user <TWS_user>
                             -password <TWS_user_password>
                             -mode <trace_mode>
   ```

   where: <*trace_mode*> is one of the following:

   active_correlation
   All communications involving the event correlator are traced.

   tws_all_jni All communications involving the jni code are traced. The jni code refers to code in shared C libraries invoked from Java. This option is used by, or under the guidance of, IBM Software Support.

tws_all All Tivoli Workload Scheduler communications are traced.

tws_alldefault Resets the trace level to the default level imposed at installation.

tws_cli All Tivoli Workload Scheduler command line communications are traced.

tws_conn All Tivoli Workload Scheduler connector communications are traced.

tws_db All Tivoli Workload Scheduler database communications are traced.

tws_info Only information messages are traced. The default value.

tws_planner All Tivoli Workload Scheduler planner communications are traced.

tws_secjni All Tivoli Workload Scheduler jni code auditing and security communications are traced. The jni code refers to code in shared C libraries invoked from Java. Only use this option under the guidance of, IBM Software Support.

tws_utils All Tivoli Workload Scheduler utility communications are traced.

tws_broker_all
All dynamic workload broker communications are traced.

tws_broker_rest
Only the communication between dynamic workload broker and the agents is traced.

tws_bridge Only the messages issued by the workload broker workstation are traced.

To reset the traces to the default value, either run the above procedure with `trace_mode` as *tws_info*, or just stop and start the server, as follows:

1. Log on to the computer where Tivoli 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 and restart the application server as described in the section on starting and stopping the application server in the *Tivoli Workload Scheduler: Administration Guide*.

To perform the same operation on your external version of WebSphere Application Server, follw the instructions in your WebSphere Application Server documentation.

Creating a core dump of the application server

If the embedded WebSphere Application Server hangs, and you decide to contact IBM Software Support for assistance, it would help the diagnosis of the problem if you could provide one or more core dumps taken during the hang. Use the following procedure to create a core dump:

1. Log on as a WebSphere Application Server administrator

2. Change to the directory:<*TWA_home*>/eWAS/profiles/twaprofile and run the script **wsadmin.sh/bat** to open the administration shell.
3. Set the *jvm* variable as follows:

   ```
   set jvm [$AdminControl completeObjectName type=JVM,process=twaserver<n>,*]
   ```

   whew <n> is the server number.
4. Run the core dump as follows:

   ```
   $AdminControl invoke $jvm dumpThreads
   ```

   This creates a core dump in the <*TWA_home*>/eWAS/profiles/twaprofile directory with the following name:

   Windows and Linux
   javacore.<*yyyymmdd*>.<*hhmmss*>.<*pid*>.txt, where yyyy = year, mm = month, dd = day, ss = second, and pid = process ID.

   UNIX javacore<*pid*>.<*time*>.txt where pid = process ID and <time> = the number of seconds since 1/1/1970.
5. Repeat step 4. The more dumps you can take, the more information is available to the support team.
6. Send the dumps, the application server log files and a detailed description of what you were doing, to IBM Software Support.

Log files for the command line client

The command line client writes its logs in the following files:

UNIX
```
<command line client install directory>/stdlist/yyyy.mm.dd/
<TWS_user>
```

Windows
```
<command line client install directory>\stdlist\yyyy.mm.dd\
<TWS_user>
```

For example, a log file created on UNIX on December 1, 2008 for the user *myUserID* where the command line client was installed in the default directory is called:

```
/opt/ibm/TWS/CLI/stdlist/2008.12.01/myUserID
```

How to activate and modify traces

To activate and manage traces for all the following components, see respectively:

Tivoli Workload Scheduler
See “Controlling Autotrace” on page 49.

Tivoli Dynamic Workload Console
See: “Activating traces in Tivoli Dynamic Workload Console” on page 12.

Tivoli Dynamic Workload Broker
See: “Activating traces for Tivoli Dynamic Workload Broker” on page 109.

WebSphere Application Server
See the *Tivoli Workload Scheduler: Administration Guide* for information about Application server - changing the trace properties.

Analyzing log files with Log Analyzer

Use Log Analyzer to display log details from the Tivoli Workload Scheduler CClog files, and compare one or more log files. It has facilities to filter log messages by a variety of criteria, reorder log messages by a variety of criteria, and search for specific messages. You can correlate two or more log files from different computers (in different time zones, if required) and select common or corresponding messages. Log Analyzer uses Eclipse technology.

Note: Various Web sites are indicated in the following procedures. These Web sites are not owned or controlled by IBM. The following steps were correct at time of writing, but might be different when you perform them. If one or more of the items discussed below is not available, contact IBM Software Support for assistance.

The information about Log Analyzer is in these sections:

Installing Eclipse and the Test and Performance Tools Platform

Eclipse is an open source community whose projects are focused on providing an extensible development platform and application frameworks for building software.

Log Analyzer requires Eclipse, version 3.1, or higher. It is available for the Windows and Linux operating systems (see Web site for full details). Tivoli Workload Scheduler uses Eclipse version 3.0 as its platform of choice for the Tivoli Information Center. However, *Eclipse, version 3.0 cannot be used for Log Analyzer* because Log Analyzer requires a higher version.

Log Analyzer also requires the *Test and Performance Tools Platform*, version 4.1, or higher.

To install Eclipse and the Test and Performance Tools Platform, follow these steps:

1. Check that you have Java run time environment (JRE) or Java development kit (JDK), version 1.4.2 or higher installed on your machine in order to run Eclipse. If you do not have the appropriate level of JRE or JDK, follow these steps:
   a. Go to www.java.com
   b. Download and install Java Standard Edition (Java SE), version 1.4.2, or higher. At time of writing, this could be found by clicking **Free Java Download** on the home page.
   c. Follow the instructions on the Web site for downloading and installing J2SE.
2. Go to the Eclipse Web site at http://www.eclipse.org/
3. Click **Downloads**.
4. Under **Third Party Distros**, click **IBM**.
5. In the description of the **Europa testing project bundle:** you should see **Eclipse Test and Performance Tools Platform (TPTP)**. This contains both the

prerequisite versions of Eclipse and the Test and Performance Tools Platform. Click **Europa testing project bundle: → Free download**.

6. Save the .zip (Windows) or .gz (UNIX) file containing the Test and Performance Tools files in a temporary directory.
7. Open the .zip or .gz and extract the files to a temporary directory.

Configuring the Log Analyzer memory

After installing Eclipse you must configure the memory usage for the Tivoli Workload Scheduler plug-in. Do the following:

1. Close Eclipse.
2. Edit the `eclipse.ini` file in the Eclipse install directory.
3. Set the following options:

   --launcher.XXMaxPermSize
   Set to:
   ```
   512m
   ```

   -vmargs
   Set to:
   ```
   -Xms100m
   -Xmx512m
   ```

   When you have finished, your file should look like this:
   ```
   -showsplash
   org.eclipse.platform
   --launcher.XXMaxPermSize
   512m
   -vmargs
   -Xms100m
   -Xmx512m
   ```
4. Start Eclipse
5. Select **Window → Preferences**
6. Expand the **Java** option
7. Click **Installed JREs**
8. Double-click the **Installed JRE** that you are using (the one in the list that is selected by a check box)
9. In the Edit JRE window, add the following to the field **Default VM Arguments**:
   ```
   -Xms100m -Xmx512m
   ```
10. Close Eclipse.

Eclipse is now ready for use with the Tivoli Workload Scheduler plug-in.

Installing and configuring the Log Analyzer plug-in

What you have installed up to now is generic software for analyzing log files. You now need to install the plug-in that Eclipse uses to read and analyze the specific Tivoli Workload Scheduler log files.

The Tivoli Workload Scheduler plug-in is located on the DVD *IBM Tivoli Workload Scheduler 8.5 Integrations, Multiplatform Multilingual* for your platform, in the following path:

```
TWS_INTEGRATION\integrations\log_analyzer\TWSLogParser.tar
```

This is a compressed archive, which contains just one file: `TWSLogParser_8.5.0.jar`. Extract this file into the `Eclipse` directory, and it is automatically placed in the `Eclipse/plugins` directory. For example, on Windows, if the location you chose to install Eclipse and the Test and Performance Tools Platform was `D:\`, you should specify to install the jar file in `D:\eclipse`

The installation of the Log Analyzer is now complete.

Upgrading Log Analyzer

If you have already installed and used Log Analyzer in a previous release of Tivoli Workload Scheduler you might decide to upgrade the analyzer to be able to use the additional facilities offered in the latest version of Eclipse, details of which can be found on the Eclipse Web site: http://www.eclipse.org/.

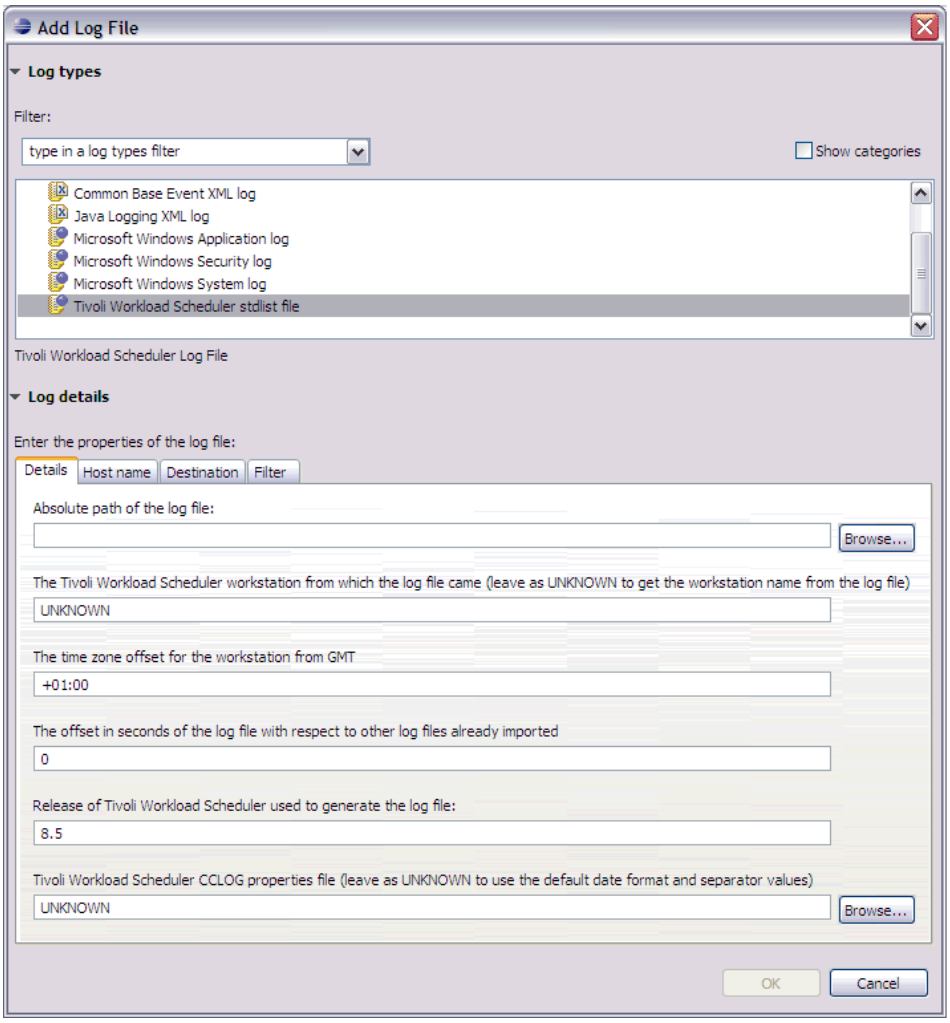
If you decide to upgrade, delete the existing Eclipse folder and all its plug-ins. Then install and configure the new version and import the log files, as described in the following sections.

If you upgrade to this version you will also need to import the new symptom catalog (formerly called a symptom database), because the format of the catalog has changed (see "Analyzing messages using a symptom catalog" on page 29) for details of the advantages of using the symptom catalog.

Adding a log file

Each log file that you want to look at or analyze must be identified to Log Analyzer, as follows:

1. Run **Eclipse**.
2. From the **File Menu** select **Import**.
3. From the list of import sources, select **Profiling and Logging → Log File**. Click **Next**.
4. On the **Import Log File** panel, select **Add**.
5. On the **Add Log File** panel, select **Tivoli Workload Scheduler stdlist file** from the list of log file types.
6. Click on the **Details** tab of the log file properties:



7. Enter or browse for the following information:

Absolute path of the log file
Enter or browse for the absolute path of the log file that you want to load. See “Understanding log and trace files” on page 7 for information about the location of log files.

The Tivoli Workload Scheduler workstation name
Leave as "UNKNOWN" and Log Analyzer fills in the information when it loads the file.

Time zone offset for the workstation from GMT
Enter the time zone offset from GMT of the workstation where the log file was recorded, in the format:

```
±hh:mm
```

The default is the time zone offset of the workstation where Log Analyzer is being run.

The offset in seconds of the log file with respect to other log files already imported
Enter any additional offset, in seconds, that this log file has from other log files already imported. The default is zero.

Release of Tivoli Workload Scheduler used to generate the log file
Enter the release of Tivoli Workload Scheduler that was running on the workstation when the log file was created. The default is 8.5.

Tivoli Workload Scheduler CCLOG properties file
Enter or browse for the path of the `TWSCCLog.properties` file (see "CCLog customization" on page 9 for the location). If the log file you want to analyze is not a CCLog file, use the properties file appropriate for the log file, or leave the field as "UNKNOWN" if you want Log Analyzer to use the default values for the date format and field separator values.

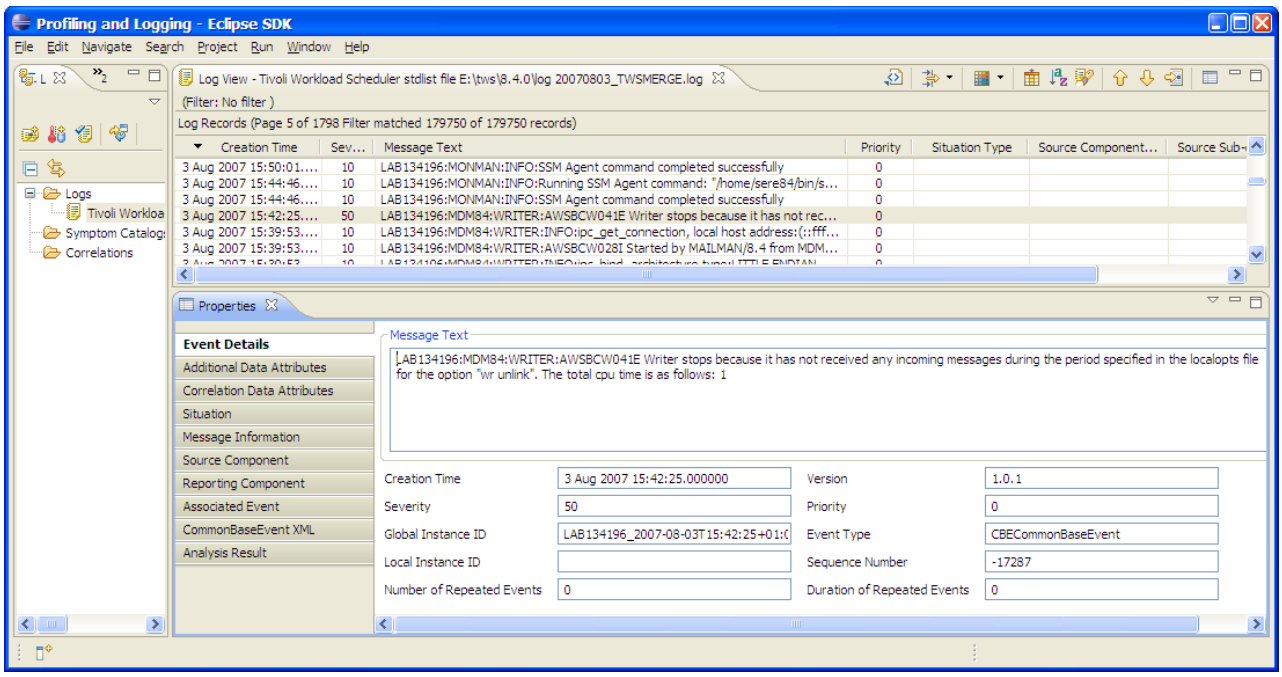
8. Click **OK**.
9. Click **Finish** on the Import Log File panel.
10. If the Confirm Perspective Switch window opens, inviting you to switch to the Profiling and Logging Perspective, click **Yes**.

Using Log Analyzer

To use Log Analyzer, run Eclipse, select a log file that you have already added (see "Adding a log file" on page 20), and use the Log Analyzer options to examine and analyze the data in the file. This section describes the available options. It contains the following topics:

Understanding the main window

After you have run **Eclipse** and the Log Analyzer window has opened with a log file already added, you see a window like the following:



The window tabs are as follows:

Log Navigator tab
: This is where your log files are listed. Correlations are created by you (see "Using correlations" on page 28), and you can work with symptom catalogs (see "Analyzing messages using a symptom catalog" on page 29)

Log View tab
: The main tab is the Log View tab. This is a list of the records in the log file. An error message with a severity of 50 has been highlighted (severities higher than the standard 10 are highlighted in yellow or red, depending on the severity, but the color disappears when you click on the message to select it.

  When a message is highlighted, its details appear in the Properties tab, below. If the Properties tab is not showing, right-click the message you want to examine and select **Properties**.

  Above the Log View tab are the icons that you use to perform the functions of Log Analyzer.

Properties tab
: This contains several panes of information about the message. Those which contain information with respect to Tivoli Workload Scheduler messages are Event Details, Additional Data Attributes, and CommonBaseEvent XML.

For general help for using Eclipse select **Help → Help Contents**.

For specific help for using Log Analyzer select **Help → Dynamic Help**

Following the log message flow

To follow the message flow, scroll down the Log Record list. Logs are listed in pages of 50 messages.

The navigation of this list is as follows:

Moving within a page
Use the scroll bars to move up and down within a page. Your keyboard's **PageUp** and **PageDown** keys move the display up and down within a single page.

Moving between pages
To move from one page to the next click the **Page-Down** icon:  icon or the **Page-Up** icon:  . Alternatively, you can jump to a particular page by clicking the **Go To Page** icon:  and entering a page number.

Locating a specific message

To locate a specific message, click the **Find Log Record** icon:  . In the Find Log Record window, click **Add** to define a search expression, by selecting a property and an operator, and entering the value or partial value for the property to search for. Wildcards can be used for the partial value.

For example, selecting the *Message Text* property with "=" (equals), and supplying a value of **AWSBCW041E** creates a search expression that, when you click Find Next, locates the first message containing the string "AWSBCW041E".

These expressions are saved automatically and permanently in the Find Log Record window. On a subsequent visit to this window you can select a search expression you have previously created or add a new one.

Sorting messages

Messages are presented by default in ascending order of *Creation Time*. If you want to change this order, follow these steps:

1. Click the **Sort Columns...** icon: 
2. Use the central arrow buttons and to move selected properties to and from the Properties list and the Selected Properties list.
3. Use the  and  buttons to move properties in the Selected Properties list into the correct sort sequence.
4. Click **OK**. The messages are redisplayed in the selected sequence.

Filtering messages

Many log files are very large, and you might only be interested in a subset of the messages in them. A filter can be applied in Log Analyzer which restricts the messages on display to those that match the filter criteria. You can do the following:

Apply an existing filter
To apply a defined filter, click the arrow beside the **Manage Filters...** icon:  to choose a filter from those you have already created yourself and the default filters (such as "All error messages"). Filters are not cumulative, so, for example, if you apply a filter for "Error messages", and then apply one that you have created for "All MAILMAN messages", you get a list of "All MAILMAN messages", not "All MAILMAN error messages".

Apply no filter
To stop the effect of the currently applied filter, click the arrow beside the **Manage Filters...** icon and select **No Filter**.

Create a new filter when no filter is in force
If no filter is in force, click the **Manage Filters...** icon to open the Filters panel and create a new filter (see "Adding a new filter" for details on the filter options available).

Create a new filter when another filter is in force
To create a new filter when another filter is in force, click the arrow beside the **Manage Filters...** icon and select the **Manage Filters...** option. From the Add/Edit/Remove Filters window click **New** (see "Adding a new filter" for details on the filter options available).

Edit a filter currently in force
If you have applied a filter and want to edit it, click the **Manage Filters...** icon to open the Filters panel and edit the filter currently in force (see "Adding a new filter" for details on the filter options available).

Edit any other filter
To edit an existing filter, click the arrow beside the **Manage Filters...** icon and select the **Manage Filters...** option. From the Add/Edit/Remove Filters window select a filter to edit, and click **Edit** (see "Adding a new filter" for details on the filter options available).

Delete (remove) a filter
To delete a filter, click the arrow beside the **Manage Filters...** icon and select the **Manage Filters...** option. From the Add/Edit/Remove Filters window click **Remove**.

Adding a new filter: To add a new filter In the Filters panel, follow this procedure:

1. Give a name to the filter.
2. Decide if you want to set either of the options on the **Standard** tab:

   Show events by severity
   Set this to select that the filter includes only specific types of message

   Show correlated log records only
   Select this if you are using a correlation, and want the filter to include only messages that are correlated. See "Using correlations" on page 28 for more details about correlations.
3. Click the **Advanced** tab.
4. Click Add to add a new filter expression. Note that you can make complex filters by creating an unlimited number of filter expressions.
5. On the Add Filter Property window, select a property and an operator, and enter the value or partial value for the property to filter for. Wildcards can be used for the partial value. These expressions are saved automatically and permanently in the Add Filter Property window when you click **OK**.
6. Click **OK** to close the Edit Filter window.
7. If the Add/Edit/Remove Filters window is open, click **OK** to close it.
8. The new filter is applied immediately. If you have a complex filter or many records, you might have to wait for the results to be visible.

For example, creating a filter expression selecting the *Message Text* property with "=" (equals), and supplying a value of **JOBMON**, and then creating a second filter expression selecting the *Creation time* property with ">" (greater than), and

supplying a value of *2008-02-08 21:53:16.38+0100* creates a filter that, when you apply it, displays only messages containing the string "JOBMON" created after the indicated date.

Creating reports

Reports of selected log details can be created in CSV, HTML, or XML formats, as follows:

1. Use the other facilities described in the above sections to select the messages for which you want to create a report.
2. Ensure that you only have the required properties selected, because the report is created using all of the selected properties. See “Managing the log message properties” for details.
3. Click the **Report ...** icon:  .
4. On the New Report panel select the **Report to be created** (CSV, HTML, or XML).
5. Decide if you want to edit the report after it is created, deselecting the **Open editor** checkbox, if not.
6. Click **Next**.
7. On the Report panel, enter the parent folder for the report, or select one of the listed folders.
8. Supply a file name for the report.
9. If you have selected an HTML Report, you can optionally click **Next** to open a panel where you select which pages of the Log Records view should be included in your report.
10. Click **Finish**. If you selected **Open editor**, the report is displayed as follows:

    CSV format
    Log Analyzer opens a window in your default application for CSV files (this might be Microsoft Excel, for example, on Windows).

    HTML format
    A pane is opened at the bottom of the Log Analyzer window.

    XML format
    A pane is opened at the bottom of the Log Analyzer window.
11. For HTML and XML reports make any changes you require. The pane does not verify the integrity of the HTML or XML after you have edited it, so any changes must be compatible with HTML format or the DTD or schema of the XML file, as appropriate.
12. If you have made any changes, when you click on the **Close** icon you are asked if you want to save the changed file.

Managing the log message properties

The message properties are not only displayed in the Property and Value pane, but also used for the search, sort, and filter actions. Some of the message properties might not be of interest to you. For example, there is a default property called *priority* that might not interest you. You can hide properties that do not interest you, as follows:

1. Click the **Choose Columns...** icon:  .
2. In the Filter Properties panel are displayed all possible properties that Log Analyzer can manage. Many of them are not properties of Tivoli Workload Scheduler log files, and can be ignored.

Use the central arrow buttons > and < to move selected properties to and from the Properties list and the Selected Properties list.

Use the Up and Down buttons to move properties in the Selected Properties list into the display order you require (Click the **Sort** buttons on either list to order the properties in alphabetical order).

3. Click OK to finish. Any properties you have selected or deselected are added to or removed from displays and selection panels and drop-downs.

Highlighting messages

Using the filters described in "Filtering messages" on page 24, you can set a highlight that automatically applies a background color to messages that match the filter in question. For example, by default, messages with a high severity (error messages) display the severity value with a red background; but using this facility you can configure Log Analyzer to display the entire message with a red background.

The following options are available:

Set highlights

Do this as follows:

1. Click the **Highlight Events...** icon:  .
2. In the Highlight Events... window select one or more defined filters by clicking their checkbox
3. For each selected filter, click the Color column and then the ellipsis button  that is displayed in the color column.
4. Select, or define and select, the color you require.
5. Click OK to finish. The chosen background color or colors will be applied to the displayed messages.

   Notes:

   a. You are using the filters only to determine the highlight – whatever filter you might have applied to the messages remains in force, but any displayed messages that match the filters have the chosen background color.

   b. If a message satisfies more than one filter, it is displayed against a black background to warn you of this duplication. To read the black text against the black background, click the message, and the text is displayed in white.

Remove highlights

To remove a highlight, open the Highlight Events window as above and deselect the appropriate filter.

Add new filters

You can add a new filter to the list of defined filters by clicking **New...** (see "Adding a new filter" on page 25 for details on the filter options available)

Edit or delete filters

You can edit or delete a filter from the list of defined filters by clicking the filter name and selecting **Edit...** or **Remove...**, as appropriate (see "Adding a new filter" on page 25 for details on the filter options available)

Show only highlighted events
To show only the highlighted events, click the arrow beside the **Highlight Events...** icon and select **Show only highlighted events**.

Using correlations

Two or more log files can be correlated, so that you can compare the messages from each. This might be useful, for example, when comparing a log from the master domain manager with a log from an agent.

To correlate log files take the following steps:

1. Ensure that you have imported the log files you want to correlate.
2. Right-click the **Correlations** folder in the **Log Navigator** tab, and select **New → Log Correlation**.
3. On the New Log Correlation panel give a name to the correlation.
4. Use the central arrow buttons > and < to move selected log files to and from the Available Logs list and the Selected Logs list.
5. Click **Next**.
6. Choose the correlation method:

   Tivoli Workload Scheduler Events Correlation
   The log files are correlated for matching Tivoli Workload Scheduler events

   Tivoli Workload Scheduler Job Execution Correlation
   The log files are correlated for matching Tivoli Workload Scheduler jobs

   Tivoli Workload Scheduler Linking Correlation
   The log files are correlated for corresponding linking and unlinking actions

   Time The log files are correlated with respect to time.

   Note that the first three correlations can only be performed on files that are in the Tivoli Workload Scheduler stdlist format.
7. Click **Finish**. The chosen log files are correlated.

The correlated log files can now be viewed in one of three ways:

Log View
This is the default. It shows the correlated messages in the first of the chosen log files. Select another log file in the **Log Navigator** pane to see the correlated messages in that file. To return to this view after working with one of the others, right-click on the **Correlation** in the **Log Navigator** pane and select **Open With → Log View**.

Log Interactions
Right-click on the **Correlation** in the **Log Navigator** pane and select **Open With → Log Interactions**. A graphic display shows how the two log files interact.

Log Thread Interactions
Right-click on the **Correlation** in the **Log Navigator** pane and select **Open With → Log Thread Interactions**. A graphic display shows how the two log files interact for individual threads.

Analyzing messages using a symptom catalog

Tivoli Workload Scheduler messages in the log file contain just the message text. To store more information about a message, or to document a course of action in respect of that message, you can create a symptom catalog, recording information in the catalog for any message that could appear in the log.

The symptom catalog is in the form of an xml file. The dtd of the xml file is simple, and can be determined by looking at the symptom catalog supplied with Tivoli Workload Scheduler.

This symptom catalog contains the message help information (explanation, system action, and operator response) for all of the messages that are logged in the Tivoli Workload Scheduler logs (from the *Maestro*, *Unison*, *Netman*, *Cluster*, and *Altinst* catalogs). To determine which messages these are, look at the beginning of each message set described in *Tivoli Workload Scheduler: Messages* – those belonging to the above-mentioned catalogs are indicated. This information is available in English, only. You can use this catalog as it is, modify the catalog, adding information pertinent to your enterprise, or create your own catalog, based on the structure of the example. Log Analyzer supports the contemporaneous presence of more than one catalog, though a message can be analyzed by only one catalog at a time.

Note: Not included are those messages logged in the log files of the application server.

This section tells you how to do the following:

- "Install the Tivoli Workload Scheduler symptom catalog"
- "Use the symptom catalog"

Install the Tivoli Workload Scheduler symptom catalog: The Tivoli Workload Scheduler symptom catalog is included in the `TWSLogParser.tar` that you have already installed. However, it needs to be separately imported into Log Analyzer, as follows:

1. Open the `TWSPLUGINS/TWSLogParser.tar`, described in "Installing and configuring the Log Analyzer plug-in" on page 19 (a zip-like utility can be used).
2. Open the `TWSLogParser_8.5.0.jar`, contained therein (a zip-like utility can be used).
3. Extract the `TWSSymptomDB.symptom` into a temporary directory.
4. Start **Eclipse**.
5. From the **File Menu** select **Import**.
6. From the list of import sources, select **Symptom Catalog File** and click **Next**.
7. On the Symptom Catalog File panel, select the **Local Host** radio button.
8. Navigate to and select the `TWSSymptomDB.symptom` file in the temporary directory created in step 3.
9. Click **Finish** on the Import Symptom Catalog File panel.

The installation of the example symptom catalog is now complete. Use a similar procedure to install your own symptom catalog, should you decide to create one.

Use the symptom catalog: If you have installed a symptom catalog (see "Install the Tivoli Workload Scheduler symptom catalog") take the following steps to see the message help for one or more messages.

1. Select the log message which you require to analyze.
2. Right-click the log message and select **Analyze** to analyze just the selected message or **Analyze All** to analyze all messages in the log file page.
3. The message or messages you have chosen to analyze are listed in the Symptom Analysis Results View.
4. Click a message in this view.
5. Click the **Properties** tab.
6. Under **Other symptom properties**, click the message number in the field **Description**.
7. If the message is present in the symptom catalog, the message number will be highlighted in the TWSSymtomDB.symptom Symptom Definitions view.
8. Expand the selection to show the **Rule** and **Effect** entries.
9. Click **Effect**. In the same panel, under **Symptom effect details**, then **Identification properties**, then **Description**, is displayed the message help.
10. Click the associated ellipsis button  to view a panel showing the Explanation, System Action and Operator Response of the message.

Maintaining an audit trail

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:

How audit works

Auditing works as follows:

- For the database, all user modifications are logged. However, the delta of the modifications, or before image and after image, are not logged. If an object is opened and saved, the action is logged even if no modification has been done.
- For the plan, all user modifications to the plan are logged. Actions are logged whether they are successful or not.

Each audit log provides audit information for one day, from 00:00:00 UTC to 23:59:59 UTC regardless of the timezone of the local workstation, but the log file is only created 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 Tivoli Workload Scheduler network. This minimizes the risk of audit failure due to network issues. The log formats are the same for both plan and database in a general sense. The logs consist of a header portion which is the same for all records, an action ID, and a section of data which 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**.

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.

Enabling the audit feature

The auditing option is enabled by setting the following two entries in the global options, using **optman**:

```
enPlanAudit = 0|1
enDbAudit = 0|1
```

A value of *1* (one) enables auditing and a value of *0* (zero) disables auditing. Auditing is disabled by default on installation of the product.

To initiate database auditing, you must shut down Tivoli Workload Scheduler completely. When you restart Tivoli Workload Scheduler, the database audit log is initiated. Plan auditing takes effect when JnextPlan is run.

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 Tivoli Workload Scheduler workstation name for which this file was created. Each workstation in the Tivoli Workload Scheduler network creates its own log.

User ID The Tivoli 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
DBCAL	Database calendar definition
DBDOMAIN	Database domain definition
DBJBSTRM	Database job stream 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

PLDOMAIN Plan domain

PLFILE Plan file

PLJBSTRM Plan job stream

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, EXPAND, 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, Tivoli 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 Tivoli 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 domain user, for Log Type **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
```

Capturing trace information at failure time

The Data Capturing Tool is a script named tws_inst_pull_info that extracts information about a product instance of Tivoli Workload Scheduler and related workstations.

With lighter prerequisites, it replaces the Metronome tool.

This script collects information that IBM Software Support can use to diagnose a problem. The Data Capturing Tool is available in Tivoli Workload Scheduler version 8.5.1 and later and can run on all the supported operating systems.

The Data Capturing Tool program is located in the <TWA_HOME>/TWS/bin directory and can be run from the UNIX or DOS prompt on the master domain manager, the backup master domain manager, or a fault-tolerant agent.

Command and parameters

The Data Capturing Tool can be used by any Tivoli Workload Scheduler user who is the owner of the Tivoli Workload Scheduler instance for which the data needs to be collected, or by other Tivoli Workload Scheduler users such as root or Windows administrator. Each Tivoli Workload Scheduler instance has an owner who is the user that is specified during the installation phase.

To use the Data Capturing Tool you must have read access to the directories `/etc` and `/etc/TWS` and, on UNIX platforms, to the `/etc/TWSRegistry.dat` file.

If you use the Data Capturing Tool on an existing Tivoli Workload Scheduler instance, a Symphony™ file is required to ensure that all files are extracted. If you are using it as a root user on a new Tivoli Workload Scheduler instance, the Symphony file is not required.

Depending on your operating system, start the Data Capturing Tool with the following commands:

Windows

```
tws_inst_pull_info.cmd -twsuser <userid> -log_dir_base <path> [-u |
-run_db2_module <y/n> | -extract_db_defs <y/n> | -date <yyyymmdd>]
```

UNIX

```
tws_inst_pull_info.sh -twsuser <userid> -log_dir_base <path> [-u |
-run_db2_module <y/n> | -extract_db_defs <y/n> | -date <yyyymmdd>]
```

where

-twuser
The Tivoli Workload Scheduler user that you specify when you install the Tivoli Workload Scheduler. This user must exist in the `/etc/TWS/TWSregistry.dat` file if the Tivoli Workload Scheduler instance already exists. This parameter is mandatory.

-log_dir_base
The base directory location where the collected data is stored. The user must have write access to the specified directory. This parameter is mandatory.

-run_db2_module
Identifies if DB2®-specific data is extracted. This operation might take some time. Valid values are `y` or `n`. Set to `y` if you want to collect DB2 specific data. This parameter is optional. The default is `n`.

-extract_db_defs
Identifies if database definitions are extracted (if the workstation is a master domain manager). Valid values are `y` or `n`. This parameter is optional. The default is `y`.

-date
Used as the base date for collected data logs. If not specified, the script uses the current date by default. Run the Data Capturing Tool as soon as a problem occurs to collect the data specific to the date and time of the problem. If the problem occurs on the current date, this option is not required. If the problem occurred earlier, then the date on which the problem occurred must be specified in the `yyyymmdd` format. Either the current date or the specified date is used to identify which files and logs are extracted. This parameter is optional.

-u Prints the message with the description of each parameter.

Tasks

Use the Data Capturing Tool to perform the following tasks:

Check that the user exists

The script verifies if the specified user exists in the `TWSRegistry.dat` file. If it does, the `<TWS_HOME>` directory used for data collection is extracted from the `TWSRegistry.dat` file. (UNIX only) If the specified user does not exist, the script verifies if the user exists in the `/etc/passwd` file. If no user exists, the script terminates.

Check the user permissions

The commands that are used during the data collection try to retain the original ownership of the files; when the script is run on Solaris platforms, the ownership of the files is not guaranteed. If the script is run by a IBM Tivoli Workload Scheduler user (for example, not the root user) the script collects the available instance data. If the user (including root or Windows Administrator) who is running the script is different from the specified Tivoli Workload Scheduler user (`-twsuser`) and exists in the Tivoli Workload Scheduler Security files, then the Tivoli Workload Scheduler Security access permission determines which data can be extracted. If the user (including root or Windows Administrator) running the script does not exist in the Tivoli Workload Scheduler Security files, then the UNIX or Windows permission determines which data is extracted.

Note:

On UNIX platforms, for a newTivoli Workload Scheduler installation, the script must be run by the root user to extract the instance data.

Some Windows security policies can affect which data is extracted.

Create the directories in which to store the collected data

The script first creates the `<log_dir_base>` directory, where `<log_dir_base>` is the value provided for the `-log_dir_base` option. Within the `<log_dir_base>` directory, the script creates the `tws_info` directory and its subdirectories `TWS_yyyymmdd_hhmmss`, where yyyy=year, mm=month, dd=day, hh=hour, mm=minute and ss=seconds.

Collect data

The script collects system and product-specific data.

Create the TAR file

On UNIX platforms, the script creates the TAR file `TWS_yyyymmdd_hhmmss.tar` and compresses it to `TWS_yyyymmdd_hhmmss.tar.Z`, or if the operating system is Linux_i386, `TWS_yyyymmdd_hhmmss.tar.gz`.

Note:

On Windows platforms, the ZIP file containing the collected data is created by the user.

Data collection

Use the Data Capturing Tool to collect the following types of data.

System-specific data

For system-specific data, the script performs the following operations:

- Extracts local CPU node information
- Extracts environment for the current IBM Tivoli Workload Scheduler instance
- Extracts `nslookup` information for local CPU
- Extracts `netstat` information for local CPU
- Extracts Services information
- Extracts the current running processes for the current Tivoli Workload Scheduler user
- Extracts the current available disk space for `%TWS_HOME%`

- Extracts the current available disk space for the `tmp` directory
- (UNIX only) Extracts the current system disk space
- (UNIX only) Extracts the current disk space of root `filesystem`
- (Solaris 10.x or above) Extracts `zonecfg` information
- (AIX® only) Copies `netsvc.conf`
- (UNIX only, except AIX) Copies the`nsswitch.*` files
- Pulls the host and services files

Tivoli Workload Scheduler-specific data

For Tivoli Workload Scheduler-specific data, the script performs the following operations:

To collect Tivoli Workload Scheduler messages:

- Generates a list of the`.msg` files
- Extracts a list of the files in the `%TWS_HOME%\ftbox` directory

To collect Tivoli Workload Scheduler information:

- Extracts information about the Tivoli Workload Scheduler instance new or existing installation
- Extracts the Tivoli Workload Scheduler Security file
- Extracts a list of the Tivoli Workload Scheduler binaries
- Extracts a list of the files in the `TWSHome` directory
- Extracts a list of the files in the `%TWS_HOME%\mozart` directory
- Extracts a list of the files in the `%TWS_HOME%\pids` directory
- Extracts a list of the files in the `%TWS_HOME%\network` directory
- Extracts a list of the files in the `%TWS_HOME%\audit\database` directory
- Extracts a list of the files in the `%TWS_HOME%\audit\plan` directory
- Extracts the database definitions to `flatfiles`
- (UNIX only) Extracts the `optman` output
- (UNIX only) Extracts `planman` "showinfo" output
- (UNIX only) Extracts the list of the `%TWS_HOME%\trace` directory
- Copies `jobmanrc.cmd` and `jobamnrc` (if it exists)
- Copies the `schedlog` files of the previous day (the option `-date` is not used)
- Copies the `schedlog` files of the day on which the problem occurred, day - 1 and day + 1 (the optoin `-date` is used)
- Copies a list of the files in `%TWS_HOME%\audit\database\${today}`
- Copies a list of the files in `%TWS_HOME%\audit\database\${yesterday}`
- Copies a list of the files in `%TWS_HOME%\audit\plan\${today}`
- Copies a list of the files in `%TWS_HOME%\audit\plan\${yesterday}`
- Copies the `BmEvents.conf` file and the event log (if `%TWS_HOME%\BmEvents.conf` exists)
- Copies the content of the `BmEvents` log file (if `%TWS_HOME%\BmEvents.conf` exists)
- Copies the `TWSREgistry.dat` file
- Copies the content of the `%TWS_HOME%\version` directory
- Copies the files for Tivoli Workload Scheduler instance `%tws_cpu%`
- (Windows only) Copies the `TWSZOSConnRegistry.dat` file

To collect Tivoli Workload Scheduler logs:

- Copies the TWSUser BATCHUP and NETMAN stdlist files for current and previous date
- Copies the TWSMERGE and NETMAN log files from the stdlist\logs directory for current and previous date
- Copies the TWSMERGE BATCHUP and NETMAN stdlist files from the stdlist\traces directory for current and previous date

To collect Tivoli Workload Scheduler files:

- Extracts a list of the files in the %TWS_HOME%\ITA directory
- Extracts a list of the files in the %TWS_HOME%\stdlist\JM directory
- Extracts a list of the files in the %TWS_HOME%\jmJobTableDir directory
- Copies the *.ini and *.log files in %TWS_HOME%\ITA
- Copies the *.out files in %TWS_HOME%\ITA
- Copies all the files in the %TWS_HOME%\stdlist\JM directory
- Copies all the files in %TWS_HOME%\jmJobTableDir

To collect data on the Tivoli Workload Scheduler methods:

- Copies the content of the TWSHome\methods directory (if it exists)
- (Windows only) Collects information about the Peoplesoft method
- Collects information about the r3batch method
- (UNIX only) Collects the R3batch picklist results

WebSphere-specific data

For WebSphere-specific data, the script performs the following operations:

- (Windows only) Extracts the list of WebSphere logs
- Extracts a list of the<WAS_DIR>/profiles
- Extracts a list of the Tivoli Workload Scheduler server files specific to WebSphere
- Copies the WebShpere logs
- Copies the Tivoli Workload Scheduler specific WebSphere logs
- Copies all the files from %WAS_PROFILE%.deleted (if it exists)
- Copies security.xml of defaultnode
- Copies all the Tivoli Workload Scheduler server files specific to WebSphere
- (On UNIX for root user only) Collects the data source properties
- (On UNIX for root user only) Collects the host properties
- (On UNIX for root user only) Collects the security properties

DB2-specific data

For DB2-specific data, the script performs the following operation:

- Collects the DB2 data by using the DB2Support tool

Data structure

Depending on the operating system, the Data Capturing Tool displays the collected data with the following directory and file structure.

Table 3. Collected data structure on UNIX

Gathered data directory structure	TWS filesystem or command	Files and listings
<root_dir>	General collector output	datagather_summary.log,TWS_<today>_files.txt NODE_<*hostname*>_TWSuser_<*twsuser*>_ Base_Date_<*yyyymmdd*>.README
<root_dir>/db2_ oracle_info	${db2user_home}/sqllib/db2dump "db2support -d <*db2db*>" output	db2diag.log db2support.zip
<root_dir>/system_info	"uname -a" output "env" output "nslookup ${local_cpu}" output "netstat -a", "netstat -rn" output "ps -ef |grep ${tws_user}" output "df -k" output "df -k /" output "df -k ${TWS_HOME}" output "df -k ${TMP_DIR}" output /etc "zonecfg list" output	cpu_node_info.txt instance_env_info.txt cpu_nslookup_info.txt cpu_netstat_info.txt ps_ef_listing.txt system_disk_available.txt root_disk_available.txt tws_home_disk_available.txt tmp_disk_available.txt hosts, services, netsvc.conf (AIX only), nsswitch.* (UNIX, except AIX) zonecfg.txt (Solaris 10.x or higher)
<root_dir>/tws_<version> _install	TWS install, upgrade log files from /tmp/TWA/tws<version> directory	*.*
<root_dir>/tws_info	${TWS_HOME}	Symphony, Sinfonia, StartUp, Jnext*, prodsked, Symnew, Jobtable, localopts, Security_file.txt (output from dumpsec), jobmanrc.txt,.jobmanrc.txt, twshome_files_ list.txt
	${TWS_HOME}/schedlog	M${today}*, M${tomorrow}*, M${yesterday}* (-date option used)M${yesterday}* (-date option not used)
	${TWS_HOME}/mozart ${TWS_HOME}/bin/* ${TWS_HOME}/ftbox ${TWS_HOME}/pids ${TWS_HOME}/network ${TWS_HOME}/audit/database ${TWS_HOME}/audit/database/${today} ${TWS_HOME}/audit/database/ ${yesterday} ${TWS_HOME}/audit/plan ${TWS_HOME}/audit/plan/${today} ${TWS_HOME}/audit/plan/${yesterday} ${TWS_HOME}/BmEvents.conf ${TWS_HOME}/BmE* Composer output on master ${TWS_REGISTRY_PATH} ${TWS_HOME}/version ${TWS_HOME}/bin/optman ${TWS_HOME}/bin/planman "showinfo" ${TWS_HOME}/trace	globalopts, mozart_dir_list.txt tws_binary_list.txt ftbox_dir_list.txt pids_dir_list.txt network_dir_list.txt audit_database_dir_list.txt audit_database_${today} audit_database_${yesterday} audit_plan_dir_list.txt audit_plan_${today} audit_plan_${yesterday} BmEvents.conf BmEvents_event_log.txt job_defs, sched_defs, cpu_defs, calendar_defs, parms_defs, resource_defs, prompt_defs, user_defs TWSRegistry.dat *.* optman_ls_info.txt planman_showinfo.txt trace_dir_image_existing_snap.txt
<root_dir>/tws_ita_files	${TWS_HOME}/ITA	*.out, ita_dir_list.txt
<root_dir>/tws_ita_bin_files	${TWS_HOME}/ITA/bin	*.ini, *.log, ita_bin_dir_list.txt
<root_dir>/tws_jobmgr_ffdc _files	N/A	--
<root_dir>/tws_jobmgr_ffdc _files/<date>	${TWS_HOME}/stdlist/JM/JOBMANAGER- FFDC/*	*.*
<root_dir>/tws_jobmgr_files	${TWS_HOME}/stdlist/JM	*.*, jobmanager_dir_list.txt
<root_dir>/tws_jobstore_files	${TWS_HOME}/jmJobTableDir/*	*.*, jobstore_dir_list.txt
<root_dir>/tws_logs	N/A	–
<root_dir>/tws_logs/stdlist	N/A	–
<root_dir>/tws_logs/stdlist /<date>	${TWS_HOME}/stdlist/<date>	twsuser and netman files from date and date-1

Table 3. Collected data structure on UNIX (continued)

Gathered data directory structure	TWS filesystem or command	Files and listings
<root_dir>/tws_logs/stdlist /logs	${TWS_HOME}/stdlist/logs	twsmerge and netman logs from date and date-1
<root_dir>/tws_logs/stdlist /traces	${TWS_HOME}/stdlist/traces	twsmerge and netman traces from data and date-1
<root_dir>/tws_methods	${TWS_HOME}/methods ./r3batch -v ./r3batch -t PL -c <cpu> -l * -j * -- "-debug -trace"	*.*, methods_dir_list.txt <cpu>_r3batch_ver.txt, r3batch version output <cpu>_r3_batch_info.txt, picklist of scheduled jobs on SAP
<root_dir>/tws_msg_files	${TWS_HOME} ${TWS_HOME}/pobox	*.msg, msg_file_listing.txt *.msg
<root_dir>/was_info	${WAS_SERVER_DIR} ${WAS_SERVER_DIR} ${WAS_PROFILE_DIR}/config/cells/ DefaultNode/security.xml "find ${WAS_DIR}/profiles" output "showDataSourceProperties.sh" output "showHostProperties.sh" output "showSecurityProperties.sh" output	${WAS_SERVER_DIR}_config_listing.txt *.* security.xml websphere_profile_home_list.txt DataSourceProperties.txt (on twsuer = root) HostProperties.txt (on twsuer = root) SecurityProperties.txt (on twsuer = root)
<root_dir>/was_info/logs	${WAS_PROFILE_DIR}/logs (WebSphere logs)	*.*
<root_dir>/was_info/ logs/<add. folders>	${WAS_DIR}/logs (TWS specific logs)	*.*

Table 4. Collected data structure on Windows

Gathered data directory structure	TWS filesystem or command	Files and listings
<root_dir>	General collector output	TWS_%today%_files.txt NODE_<hostname>_TWSuser_<twsuser>_ Base_Date_<yyyymmdd>.README
<root_dir>/db2_oracle_info	"db2support -d *<db2db>*" output	db2support.zip
<root_dir>/system_ info	"netstat -abenoprsv" output "echo %COMPUTERNAME%" output "nslookup %local_cpu%" output %windir%\System32\drivers\etc\hosts "set" output (sc qc tws_maestro_%tws_user% output sc qc tws_netman_%tws_user% output sc qc tws_tokensrv_%tws_user% output) %windir%\System32\drivers\etc\services dir /w "%TMP_DIR%" dir /w "%TWS_HOME%" ntprocinfo.exe -v|findstr /I /c:%TWS_HOME%	cpu_netstat_info.txt cpu_node_info.txt cpu_nslookup_info.txt hosts instance_env_info.txt local_services_info.txt services tmp_disk_available.txt tws_home_disk_available.txt tws_process_listing.txt

Table 4. Collected data structure on Windows (continued)

Gathered data directory structure	TWS filesystem or command	Files and listings
<root_dir>/tws_info	%TWS_HOME%\Symphony %TWS_HOME%\Sinfonia %TWS_HOME%\StartUp.cmd %TWS_HOME%\Jnext*.* %TWS_HOME%\prodsked %TWS_HOME%\Symnew %TWS_HOME%\Jobtable %TWS_HOME%\schedlog %TWS_HOME%\localopts %TWS_HOME%\mozart\globalopts %TWS_HOME%\bin\dumpsec %TWS_HOME%\jobmanrc.cmd %TWS_HOME%\djobmanrc.cmd dir %TWS_HOME%* dir %TWS_HOME%\bin* dir %TWS_HOME%\mozart* dir %TWS_HOME%\pids* dir %TWS_HOME%\network* dir %TWS_HOME%\audit\database* %TWS_HOME%\audit\database\%today% %TWS_HOME%\audit\database\%yesterday% %TWS_HOME%\audit\plan %TWS_HOME%\audit\plan\%today% %TWS_HOME%\audit\plan\%yesterday% %TWS_HOME%\BmEvents.conf %TWS_HOME%\BmE* Composer output on master %WINDIR%\system32\TWSRegistry.dat %WINDIR%\system32\TWSZOSConnRegistry. datxcopy /S "%TWS_HOME%\version	Symphony Sinfonia StartUp.cmd Jnext*.* prodsked Symnew Jobtable M%today%*, M%tomorrow%*, M%yesterday%* (-date option used) M%yesterday% (-date option not used) localopts globalopts Security_file.txt jobmanrc.txt djobmanrc.txt twshome_files_list.txt tws_binary_list.txt mozart_dir_list.txt pids_dir_list.txt network_dir_list.txt audit_database_dir_list.txt audit_database_%today% audit_database_%yesterday% audit_plan_dir_list.txt audit_plan_%today% audit_plan_%yesterday% BmEvents.conf BmEvents_event_log.txt job_defs, sched_defs, cpu_defs, calendar_defs, parms_defs, resource_defs, prompt_defs, user_defs TWSRegistry.dat TWSZOSConnRegistry.dat *.*
<root_dir>/tws_ <version>_install	xcopy /S "%TEMP%\tws%TWS_VMR%" xcopy /S "%TEMP%\tws%TWS_VMR%fixpack"	*.* *.*
<root_dir>/tws_ita_ files	dir %TWS_HOME%\ITA %TWS_HOME%\ITA*.ini %TWS_HOME%\ITA*.log %TWS_HOME%\ITA*.out	ita_dir_list.txt *.ini *.log *.out,
<root_dir>/tws_ jobmgr_ffdc_files	%TWS_HOME%\stdlist\JM\JOBMANAGER-FFDC*	*.*
<root_dir>/tws_ jobmgr_files	dir %TWS_HOME%\stdlist\JM %TWS_HOME%\stdlist\JM*	jobmanager_dir_list.txt *.*
<root_dir>/tws_ jobstore_files	dir %TWS_HOME%\jmJobTableDir %TWS_HOME%\jmJobTableDir*	jobstore_dir_list.txt *.*
<root_dir>/tws_ logs	%TWS_HOME%\stdlist/<date> %TWS_HOME%\stdlist/logs %TWS_HOME%\stdlist/traces	twsuser, netman and batchup files from date and date-1 twsmerge and netman logs from date and date-1 twsmerge and netman traces from data and date-1
<root_dir>/tws_ methods	%TWS_HOME%\methods* echo %CMDEXTVERSION% (PeopleSoft method) psagent.exe -v (PeopleSoft method) r3batch -v (SAP method)	*.* CMDEXTVERSION.txt psagent_exe_v.txt r3batch_ver.txt
<root_dir>/tws_msg_ files	"%TWS_HOME%*.msg" "%TWS_HOME%\pobox*.msg" dir "%TWS_HOME%\ftbox*"	*.msg, msg_file_listing.txt *.msg ftbox_dir_list.txt
<root_dir>/tws_methods	${TWS_HOME}/methods ./r3batch -v ./r3batch -t PL -c <cpu> -l * -j * -- "-debug -trace"	*.*, methods_dir_list.txt <cpu>_r3batch_ver.txt, r3batch version output <cpu>_r3_batch_info.txt, picklist of scheduled jobs on SAP

Table 4. Collected data structure on Windows (continued)

Gathered data directory structure	TWS filesystem or command	Files and listings
<root_dir>/was_info	dir "%WAS_SERVER%* dir "%WAS_HOME%\profiles %WAS_PROFILE%\config\cells\DefaultNode\ security.xml @cmd /C "%WAS_TOOLS%\showDataSource Properties.bat" @cmd /C "%WAS_TOOLS%\showHost Properties.bat" @cmd /C "%WAS_TOOLS%\showSecurity Properties.bat"	%WAS_SERVER%_config_listing.txt websphere_profile_home_list.txt security.xml showDataSourceProperties.txt showHostProperties.txt showSecurityProperties.txt
<root_dir>/was_info/ %WAS_SERVER%_config_files	%WAS_SERVER%*	*.*
<root_dir>/was_info/ MAIN_WAS_LOGS	xcopy /S "%WAS_HOME%\logs" (WebSphere logs)	*.*
<root_dir>/was_info/ %WAS_PROFILE%_logs	xcopy /S "%WAS_PROFILE%\logs" (TWS specific logs)	*.*
<root_dir>/was_info/ %WAS_PROFILE%.deleted	xcopy /S "%WAS_HOME%\profiles\ %WAS_PROFILE%.deleted"	*.*

Keeping a trace record for the main components

Tivoli Workload Scheduler is supplied with Autotrace, a built-in trace mechanism that you can use, in conjunction with IBM Software Support, to obtain trace information to be used for problem solving. This section describes how to use Autotrace, in the following topics:

Autotrace overview

Autotrace is a built-in flight-recorder-style trace mechanism that logs all activities performed by the Tivoli Workload Scheduler processes in a configurable memory buffer. In case of product failure or unexpected behavior, this feature can be extremely effective in helping to find the cause of the problem and provide a solution quickly. Autotrace is a third-party product which is licensed for use with Tivoli Workload Scheduler.

Autotrace is used in conjunction with IBM Software Support. When a problem is encountered, you submit the commands that write a snapshot of the buffer to a file, and then send the file to IBM Software Support for analysis. The information written to the trace file consists of the input and output parameters for each process call.

Autotrace uses multiple memory channels to manage the tracing of multiple components. Each Tivoli Workload Scheduler component (process) is associated with a unique *channel number*. You use this channel number to tell Autotrace which component's trace to take a snapshot of.

The tracing system has a minimal impact on system and disk accessing performance, because it is fully handled in memory. It is automatically started when the Tivoli Workload Scheduler engine starts, so no further action is required.

Autotrace

This feature is available with Tivoli Workload Scheduler on the following operating systems:

- AIX
- HP-UX
- Solaris Operating Environment
- Microsoft Windows, 2000, XP, 2003
- Linux for Intel

The feature is not available on the following operating systems:

- Linux 64 bit
- Windows 64 Bit
- HP-UX Itanium
- Solaris Operating Environment - Opteron (when configured as master domain manager)

Autotrace uses a dynamic link library named `libatrc`. This library is located in the *<TWA_home>*`/TWS/trace` directory, and on UNIX can also be found in the `/usr/Tivoli/TWS/TKG/3.1.5/lib`.

Autotrace consists of two elements:

On the workstation
A trace collector, enabled and controlled by you.

At IBM Software Support
A trace analyzer operated by the IBM Software Support staff.

When you report a problem to IBM Software Support, you could be asked to activate Autotrace for one or more processes and to perform some or all of the following steps:

1. Edit the `config` file as explained in the next section to:
   - identify or add the process you want to trace.
   - assign a free channel to the selected process if a channel is not already associated.
   - change the buffer size of the selected channel as instructed by IBM Software Support.

   For example, if you need to trace the `optman` process for which there is no default channel assigned, add the following lines:

   ```
   process optman:
      chan 16
   channel 16:
      size 10m
   ```

2. Save the `config` file.
3. Stop Tivoli Workload Scheduler processes.
4. Optionally remove any configuration or trace data in memory for the selected channel using the **actl remove** command followed by the specific channel number.
5. Re-initialize Autotrace by running the **actl init** command.
6. Start Tivoli Workload Scheduler processes
7. Optionally create a snapshot of the channel for the selected process using the **actl snap** command followed by the channel number.

8. Send the output files to IBM Software Support.
9. When the problem is resolved, reset the situation by repeating all the steps, making sure you delete the lines you added in step 1 on page 44

For some usage examples, see Autotrace tasks (example scenarios).

Configuring Autotrace

Autotrace is configured by editing the following files: `product` and `config`; which can be found in the directory <*TWA_home*>/TWS/trace/.

Autotrace is designed as a multiproduct tracer, the same instance of the software and its configuration files controlling more than one product. However, Tivoli Workload Scheduler uses a dedicated instance of Autotrace. If you have other products on a workstation that use Autotrace, they do not share either the software instance or the configuration files with the version installed with Tivoli Workload Scheduler.

Note: There might be circumstances where IBM Software Support ask you to share a single set of Autotrace configuration files between all Tivoli products. In this case, be aware that not only do different Tivoli products use different product ids, but also different ranges of channel numbers.

Configure products

The `product` file lets you configure the following aspects:

Assign a name to a product ID

Autotrace-enabled products are compiled with a product ID that is recognized by Autotrace. Tivoli Workload Scheduler is supplied with a product ID = “49420002”. The product ID must not be changed in this file.

You can associate a name with the product ID so that the product can be referred to using the name instead of the product ID, when using commands. The default name for Tivoli Workload Scheduler is “TWS”, but it can be changed to a name of your choice, by editing this file.

An example of the file as shipped, is as follows:

```
# AutoTrace product install database
#
# This file contains a partial list of products using AutoTrace.
# If the product is instrumented, but is not installed into the control
# channel via `atctl init', it is NOT traced.
#
# The file format is flat ascii, one product record per line.
# There are three fields, separated by whitespace (space and tabs);
# the first is the product id expressed as a hex value, the second
# is a short version of the product's name, and the last field is
# the product's home directory.  All fields but the first are optional.
# A field value of dash '-' is interpreted as an empty field.
#
# As an example, the following record defines product 49420002,
# with no home directory, using the alias `TWS' as a shorthand for
# the full product name `IBM Tivoli Workload Scheduler'.
#
# Product Name Home
  49420002 TWS -
```

Configure channels

The `config` file is provided by IBM pre-configured with the channels that are associated with each of the Tivoli Workload Scheduler processes, and the default parameters associated with each channel. The principal channels used are described in Table 5:

Table 5. Autotrace channels used by Tivoli Workload Scheduler

Process ([1])	Channel	Size
batchman	13	1m (1MB)
conman	15	1m (1MB)
jobman	14	1m (1MB)
JOBMAN		
JOBMON		
mailman	12	1m (1MB)
netman	11	512k (512KB - default)
All processes ([2])	10	1m (1MB)

Notes:

1. Only the most important processes are detailed here. All Tivoli Workload Scheduler processes are linked to a unique channel number.
2. Channel 10 captures the traces for all processes, including those allocated to the other specific channels.

The file lets you configure the following aspects for each channel:

Identify the trace channel (chan)

In principle, Autotrace is capable of monitoring more than one product or process simultaneously, by assigning the product ID or the name of the process being monitored to one of up to 255 trace channels. However, the autosnapshot feature of Tivoli Workload Scheduler (see "Autosnapshot (ffdc)" on page 52) relies on several of its processes having specific channels assigned to them. Thus, do not change the assigned channels, unless instructed to do so by IBM Software Support.

Set the channel buffer size (size)

Autotrace saves data into a trace channel buffer in memory, the size of which can be configured, from 100K to 256M. When the channel is full, the oldest data is overwritten first.

The size is determined by the *size* attribute of the appropriate *channel* stanza. If no *channel* stanza is present, or the *channel* stanza does not have a *size* attribute, the size is determined by the *size* attribute of the *default* stanza.

To change the amount of memory used by a channel, change or add the *size* attribute of the appropriate stanza.

For example, if you have determined that the buffer size for Netman is too small, and needs to be four times as big, proceed as follows (refer to the example file, below):

1. Identify the channel that Netman is using. In this case it is channel 11.
2. Look for a channel stanza for channel 11. If you do not find one, it means that this channel uses the values in the default stanza.
3. Change the existing channel 11 stanza, or create a channel stanza for channel 11.
4. Set the value of the size attribute in the channel stanza to "2m", which is four times the "512k" value of the default size.

Note: If IBM Software Support ask you to snap a trace for a particular problem, they normally ask you first to set the channel size to 10MB to ensure that the required information is captured. Do not forget to reset the channel to its original level after the problem has been solved.

Significant impact on memory usage and performance: The values you set here impact the amount of shared memory that Tivoli Workload Scheduler uses. Autotrace allocates shared memory for every channel indicated in the config file, and for the memory quantity indicated in the config file. Allocating too much memory to these channels causes excessive paging, with a consequent impact on workstation performance.

Set process name length (nlen)

By default, Autotrace uses the first eight bytes of the process name when writing to the trace channel. You can change this to any other multiple of four up to 64, by editing the *nlen* attribute of the *default* stanza, or a *channel* stanza.

Do not change any other parameters in this file.

An example of the file as supplied with the product, is as follows:

Autotrace

```
# AutoTrace configuration file
#
# This file is used to customize trace configurations.
# It is installed (encoded into AutoTrace control channel)
# by `atctl init' (sync) or `atctl config replace'.
# Processes only examine the installed configuration when
# they start; existing processes ignore any changes.
# The installed configuration may be displayed using `atctl config'.
#
# The file format is a relaxed stanza.  Stanzas may be defined
# in any order desired.  There are four keywords that introduce
# a new stanza; most take an argument to restrict their effect:
# default
# product $id       # 32-bit value or a product name
# process $name # process name as invoked
# channel $number # 1..255, inclusive
#
# Each stanza accepts a set of attributes, used to define the
# configuration:
# chan $number # 1..255, inclusive
# size $number # 100k..256M, inclusive
# nlen $number # 0..32, inclusive, multiple of 4 (0,4,8,12,...)
# auth $number # 0 or 1
#
#
 default:
  size 512k
  nlen 8
 product TWS:
  chan 10
 channel 10:
  size 1m
 process netman:
   chan 11
 process mailman:
   chan 12
 channel 12:
  size 1m
 process batchman:
   chan 13
 channel 13:
  size 1m
 process jobman:
   chan 14
 process JOBMAN:
   chan 14
 process JOBMON:
   chan 14
 channel 14:
  size 1m
 process conman:
   chan 15
 channel 15:
  size 1m
```

Verify path variables on UNIX

On UNIX operating systems, before running any Autotrace commands, you must verify that the path variables have been correctly set, as follows:

All UNIX operating systems other than HP-UX (32-bit and 64 bit)

```
ATTRACELIB=/usr/Tivoli/TWS/TKG/3.1.5/lib
```

HP-UX (32-bit and 64 bit)

```
ATTRACELIB=/usr/Tivoli/TWS/TKG/3.1.5.1/lib
```

Controlling Autotrace

The trace collector is controlled by the following command:

atctl

Controls the Autotrace trace collector.

Format: atctl info

atctl init <*TWA_home*>**/TWS/trace**

atctl off TWS { **all** | <*process_name*> }

atctl on TWS { **all** | <*process_name*> }

atctl product

atctl remove { <*channel_number*> | **all** }

atctl reset <*channel_number*>

atctl resume <*channel_number*>

atctl snap <*channel_number*> [<*snap_file*>]

atctl suspend <*channel_number*>

atctl trace TWS

atctl version

Purpose: Use this command to initialize Autotrace, activate the trace for Tivoli Workload Scheduler, view the trace, and copy a snapshot into a file. The options shown here are a subset of those supplied with Autotrace.

Parameters:

info
Sends the status of all active channels of the trace to STDOUT. The information comprises:

- Number of the channel
- Size of the channel
- Version of the library that created the channel
- Number of records that have been written to the channel (NLIFE)

Note: This might not be the number of records currently in the channel, because the number reported was only corrected at the moment the **atctl** command was processed, after which the channel might change if functions write to it.

init <*TWA_home*>**/TWS/trace**
Initializes Autotrace, using the configuration file (`config`) found in the directory defined as the second argument, in this case <*TWA_home*>`/TWS/trace`, where <*TWA_home*> is the installation directory of Tivoli Workload Scheduler. If a path name in a parameter includes spaces, the parameter must be surrounded by double quotes.

Note: On the Windows 2003 Server, Autotrace cannot be initialized using any remote access method. Instead it must be initialized locally. You might need to shutdown all Tivoli Workload Scheduler processes, reboot the computer and re-initialize Autotrace before starting the Tivoli Workload Scheduler processes again.

off TWS { all | <*process_name***> }**
Deactivates the trace on all or selected processes for the specified product (in this case Tivoli Workload Scheduler).

on TWS { all | <*process_name***> }**
Activates the trace for the product defined in the second argument (in this case Tivoli Workload Scheduler) and for the process or processes defined in the third argument. You normally define **all**, but you might be requested by IBM Software Support to input one or more specific process IDs here, which would be supplied to you.

product
Displays the following information for each product initialized:

- Product ID
- Product name
- Home directory

atctl obtains "product ID" and "product name" from the `product` file.

remove { <*channel_number*> | **all**
Removes any configuration or trace data in memory for the selected channel or all channels. It must be followed by an **atctl init** command.

reset <*channel_number*>
Resets the trace channel, erasing all the trace data that it contains.

resume <*channel_number*>
Resumes tracing for a given trace channel. Used after **atctl suspend** has been used.

snap <*channel_number*> **[**<*snap_file*>**]**
Takes a snapshot of the contents of the indicated trace channel and copies it to a snap file. The arguments are as follows:

<channel_number>
Obtain this value from IBM Software Support, or if you know which process you want to snap, obtain the channel number for that process from the `config` file.

[*<snap_file>*]
The name (and optionally the path) of the snap file. If this argument is omitted, the snap file is called `snap<channel_number>_<sequential_number>.at`, where:

<channel_number>
The channel number of the snapped channel.

<sequential_number>
A sequential number to distinguish the snaps for the selected channel.

The snap file is created by default in the `<TWA_home>/TWS/trace` directory. If a path is specified for this file, and it contains spaces, the whole parameter must be enclosed in double quotes.

Notes:

1. The snapshot must be taken as soon as possible after the event that caused the problem occurred.
2. If IBM Software Support ask you to snap a trace for a particular problem, they normally ask you first to set the channel size to 10MB to ensure that the required information is captured; do not forget to reset the channel to its original level after the problem has been solved.

Taking the snap resets the channel to zero so that more information can be accumulated.

suspend *<channel_number>*
Suspends tracing for a given trace channel. Use **atctl resume** to continue tracing.

trace TWS
Lists all the trace IDS which are active for product, in this case Tivoli Workload Scheduler.

version
Displays the version of Autotrace. Tivoli Workload Scheduler installs Autotrace version 3.1.5, though note that if you run this command it only shows the version and release level, for example, *3.1*.

Examples:

Example 1
Enter the following to initialize the trace:

```
atctl init "C:\Program Files\TWS\trace"
```

Example 2
To activate the trace for all Tivoli Workload Scheduler processes, use the following:

```
atctl on TWS all
```

Example 3

To deactivate the trace for Netman (channel 11), use the following:

```
atctl off TWS 11
```

Example 4

To send status information about the trace to STDOUT, use the following:

```
atctl info
```

Example 5

To create a snapshot of Netman (channel 11), giving the snapshot file a name of your choice, use the following:

```
atctl snap 11 my_snap_file
```

Example 6

To create a snapshot of Netman (channel 11), using the default snapshot file name, use the following:

```
atctl snap 11
```

Autosnapshot (ffdc)

In addition to providing a user-driven snapshot facility, Autotrace is also used by Tivoli Workload Scheduler to provide a First Failure Data Capture (FFDC) facility. FFDC is a process by which a product captures trace information at the moment that it detects that another process has failed. The Tivoli Workload Scheduler mailman, batchman, and jobman processes perform limited monitoring of the status of each other, as follows:

- If batchman detects that jobman has failed, it performs a snapshot of channel 14, which contains the traces of jobman, JOBMAN, and JOBMON.
- If mailman detects that batchman has failed, it performs a snapshot of channel 13, which contains the traces of batchman.

The autosnapshot trace is written to the following file:

<*TWA_home*>/TWS/stdlist/traces/TWS-snap-CH<*channel_number*>-<*unique_number*>.at,

where:

<*channel_number*>
The channel number of the snapped channel.

<*unique_number*>
A unique sequential number.

Autotrace tasks (example scenarios)

The following are some examples of common tasks you might be asked to perform by IBM Software Support:

Increase the channel size of batchman to 10MB and take a snap

You are being asked to increase the channel size of a component, for example batchman, to 10MB and then to take a snapshot. The steps are as follows:

1. Edit the `config` file

2. Locate the entry "process Batchman:"
3. Identify the channel number for batchman (usually 13)
4. Locate the entry for the channel (for example "channel 13")
5. Change the size parameter that follows this entry to read "size 10m"
6. Save the `config` file
7. Stop Tivoli Workload Scheduler, in particular batchman, as follows:
   a. **conman stop;wait**
   b. **conman shut;wait**
   c. On Windows only; **shutdown**
   d. Stop the SSM agent, as follows:
      - On Windows, stop the Windows service: Tivoli Workload Scheduler SSM Agent (for <*TWS_user*>).
      - On UNIX, run **stopmon**.
8. Remove the channel associated with batchman, as follows:

   atctl remove 13
9. Re-initialize the channel, as follows:

   atctl init On UNIX operating systems this step can be omitted.
10. Start Tivoli Workload Scheduler (which re-initializes Autotrace for TWS), as follows:
    a. **StartUp**
    b. **conman start** (run as the TWS_user)
11. Take a snap of channel 13 (batchman) as follows:

    atctl snap 13
12. Send the snap file (which has been given the default name of `snap13_<unique_number>.at`) to IBM Software Support
13. When the problem is resolved, reset the situation as it was before, repeating steps 1 on page 52 to 10, but setting the size parameter back to "1m"

Add the process "`maestro_plan`" as channel 17 to autotrace with a 20MB channel

You are being asked to identify to Autotrace a Tivoli Workload Scheduler process that is not currently being traced, and to assign it a 20MB channel size. The steps are as follows:

1. Edit the `config` file
2. Add the following entries:

```
process maestro_plan:
   chan 17
channel 17:
   size 20m
```

3. Save the `config` file
4. Stop Tivoli Workload Scheduler, in particular batchman, as follows:

   conman stop
5. Re-initialize Autotrace, as follows:

   atctl init

   On UNIX operating systems this step can be omitted.

6. Start Tivoli Workload Scheduler (which re-initializes Autotrace for TWS), as follows:

   conman stop

7. Do not forget that when the problem is resolved, you must reset the situation to how it was before, repeating all of these steps, but deleting the four lines you entered in step 2 on page 53

Troubleshooting Autotrace

Several situations have been identified that could require your intervention with respect to Autotrace:

Autotrace does not initialize on Windows 2003 Server

On the Windows 2003 Server, Autotrace cannot be initialized using any remote access method. If you try and initialize it and take a snapshot, the following message is displayed:

```
atctl:autotrace not initialized
```

Instead it must be initialized locally. If initializing it locally does not work, you must shutdown all Tivoli Workload Scheduler processes, reboot the computer and then re-initialize Autotrace before starting the Tivoli Workload Scheduler processes again.

Autotrace causing conman start to fail on UNIX systems

On UNIX systems that support autotrace, the **conman start** command, run as the *root* user, gives an error saying that it is not possible to load the `atrc library`.

Cause and solution:

This is a product limitation. Possible workarounds are:

- Use the StartUp utility instead of **conman start**
- Run **conman start** as the *TWS_user*

Autotrace using too much memory

If you have determined that Tivoli Workload Scheduler is using too much memory, causing excessive memory paging, with a consequent impact on the performance of the workstation, you can reduce the amount of memory used to monitor each channel (*size* parameter in `config` file). See "Configuring Autotrace" on page 45 for how this is done.

Note: Be aware that reducing the amount of memory means that the number of trace messages that are available for a snapshot is also reduced; thus, a snapshot might not contain all of the trace information that is required by IBM Software Support to understand a particular problem.

Autotrace impacting performance in mixed environment

When running many instances of Tivoli Workload Scheduler of different versions (8.5, 8.4, and 8.3, for example) on the same system, you might experience a significant performance degradation on all versions. This is due to conflicting autotrace interactions. To solve the problem upgrade the back-level instances of Tivoli Workload Scheduler.

Autotrace impacting performance

If you feel that the performance of a workstation is poor, and you believe, or IBM Software Support have suggested, that Autotrace might be affecting

the performance of the workstation, you can change Autotrace to monitor fewer functions. See "Changing the number of functions monitored" for how.

Note: Be aware that reducing the number of functions traced means that a snapshot might not be tracing the function that has caused a particular problem.

Autotrace not tracing sufficient functions for a problem to be identified

If you have experienced a problem, and IBM Software Support is unable to find the cause from the information provided by your Autotrace snapshot, they might ask you to increase the number of functions monitored, so as to provide trace information at a greater depth. See "Changing the number of functions monitored" for how.

Note: Be aware that increasing the number of functions traced might impact workstation performance.

Autotrace error received at initialization

On UNIX platforms an error message similar to the following is received at startup:

```
Sep 13 21:47:24 kerokero root: init_trace before init
Sep 13 21:47:24 localhost 1126615644 autotrace[14890]: W: trace:
no map for prod 49420002, idf 01002c46, lal 04490008
```

Cause and solution: The key phrase in this message is `init_trace before init`, which indicates that there is a problem with the initialization of Autotrace. This problem only occurs rarely, but if it does, Autotrace does not work.

To fix the problem, edit the Tivoli Workload Scheduler **Startup** script, moving the **init_trace** command so that it precedes the **netman** invocation, instead of following it. Save the amended script and the problem does not recur.

Changing the number of functions monitored

Problems of poor performance or inadequate trace monitoring are resolved by changing the Autotrace parameters to monitor a different subset of Tivoli Workload Scheduler functions. To do this follow these steps:

1. Locate and edit the `init_trace` file. This is the initialization file that is used by Autotrace when starting up.

   In this file you must change only the variable that indicates the file identifying the subset of Tivoli Workload Scheduler functions that are to be monitored.

   The file can be found in the *<TWA_home>*`/TWS/trace/` directory.

2. In the file, locate the string `ffdc_out`. This forms part of the name of a file that the initialization process uses. The possible values for the file name are as follows:

   `ffdc_out_15`
   This is the default. The value "15" indicates that all except 15% of the functions are monitored.

   `ffdc_out_30`
   The value "30" indicates that all except 30% of the functions are monitored. Modify the `init_trace` file to point to this file if you are experiencing performance problems.

`ffdc_out_5`
: The value "5" indicates that all except five percent of the functions are monitored. Modify the `init_trace` file to point to this file if you need to monitor more functions to locate the source of a problem.

3. Change the value to one of these three, according to your requirements. No other values are available.
4. Save the `init_trace` file.
5. Re-initialize Autotrace using the **atctl init** <*TWA_home*>**/TWS/trace** command.
6. Check if the change has resolved the problem.

Note: You must not attempt to directly modify the `ffdc_out_5`, `ffdc_out_15`, or `ffdc_out_30` files themselves.

Chapter 3. Troubleshooting performance issues

The performance of Tivoli Workload Scheduler can depend on many factors. Preventing performance problems is at least as important as resolving problems that occur. For this reason, all discussion of performance issues has been placed together in the chapter on performance in the *Tivoli Workload Scheduler: Administration Guide*.

Chapter 4. Troubleshooting networks

This section describes how to resolve problems in the Tivoli Workload Scheduler network. It covers the following topics:

Network recovery

Several types of problems might make it necessary to follow network recovery procedures. These include:

- Initialization problems that prevent agents and domain managers from starting properly at the start of a new production period. See "Initialization problems."
- Network link problems that prevent agents from communicating with their domain managers. See "Network link problems" on page 60.
- Loss of a domain manager, which requires switching to a backup. See "Losing a domain manager" on page 61.
- Loss of a master domain manager, which is more serious, and requires switching to a backup or other more involved recovery steps. See "Losing a master domain manager" on page 61.

Note: In all cases, a problem with a domain manager affects all of its agents and subordinate domain managers.

Initialization problems

Initialization problems can occur when Tivoli Workload Scheduler is started for a new production period. This can be caused by having Tivoli Workload Scheduler processes running on an agent or domain manager from the previous production period or a previous Tivoli Workload Scheduler run. To initialize the agent or domain manager in this situation, do the following:

1. For a domain manager, log into the parent domain manager or the master domain manager. For an agent, log into the agent domain manager, the parent domain manager, or the master domain manager.
2. Run the Console Manager and issue a **stop** command for the affected agent.
3. Run a **link** command for the affected agent. This initializes and starts the agent.

If these actions fail to work, check to see if netman is running on the affected agent. If not, issue the **StartUp** command locally and then issue a **link** command from its domain manager.

If there are severe network problems preventing the normal distribution of the new Symphony file, a fault-tolerant agent or subordinate domain manager can be run as a standalone system, provided the following conditions are met:

- The Sinfonia file was generated on the master domain manager after the network problem occurred, and so has never been transferred to the agent or domain manager
- You have some other method, such as a physical file transfer or FTP to transfer the new Sinfonia file from the master domain manager to the agent or subordinate domain manager

- The master domain manager and the agent or subordinate domain manager have the same processor architecture

If these conditions are met, do the following

1. Stop the agent or domain manager
2. Delete the <*TWA_home*>/TWS/Symphony file on the agent or domain manager
3. Copy the file <*TWA_home*>/TWS/Sinfonia from the master domain manager to the <*TWA_home*>/TWS directory on the agent or domain manager.
4. Rename the copied file <*TWA_home*>/TWS/Symphony
5. Run **StartUp** to start the agent or domain manager.

   Any inter-workstation dependencies must be resolved locally using appropriate console manager commands, such as **Delete Dependency** and **Release**.

Network link problems

Tivoli Workload Scheduler has a high degree of fault tolerance in the event of a communications problem. Each fault-tolerant agent has its own copy of the Symphony file, containing the production period's processing. When link failures occur, they continue processing using their own copies of the Symphony file. Any inter-workstation dependencies, however, must be resolved locally using appropriate console manager commands: **deldep** and **release**, for example.

While a link is down, any messages destined for a non-communicating workstations are stored by the sending workstations in the <*TWA_home*>/TWS/pobox directory, in files named <*workstation*>.msg. When the links are restored, the workstations begin sending their stored messages. If the links to a domain manager are down for an extended period of time, it might be necessary to switch to a backup (see *Tivoli Workload Scheduler: Administration Guide*).

Notes:

1. The **conman submit job** and **submit schedule** commands can be issued on an agent that cannot communicate with its domain manager, provided that you configure (and they can make) a direct HTTP connection to the master domain manager. This is configured using the *conman connection* options in the localopts file, or the corresponding options in the useropts file for the user (see the *Tivoli Workload Scheduler: Administration Guide* for details).

   However, all events have to pass through the domain manager, so although jobs and job streams can be submitted, their progress can only be monitored locally, not at the master domain manager. It is thus always important to attempt to correct the link problem as soon as possible.
2. If the link to a standard agent workstation is lost, there is no temporary recovery option available, because standard agents are hosted by their domain managers. In networks with a large number of standard agents, you can choose to switch to a backup domain manager.

Troubleshooting a network link problem

When an agent link fails it is important to know if the problem is caused by your network or by Tivoli Workload Scheduler. The following procedure is run from the master domain manager to help you to determine which:

1. Try using **telnet** to access the agent: **telnet** <*node*>**:**<*port*>
2. Try using **ping** to access the agent: **ping** <*node*>**:**<*port*>
3. Run **nslookup** for the agent and the master domain manager from both, and check that the information on each system is the same from each system
4. Run **netstat -a |grep <port>** and check if any FIN_WAIT_2 states exist

5. Verify that the port number of the master domain manager matches the entry for "`nm port`" in the `localopts` file of the master domain manager
6. Verify that the port number of theagent matches the entry for "`nm port`" in the `localopts` file of the agent
7. Check the `netman` and `TWSMerge` logs on both the master domain manager and the agent, for errors.

Notes:

1. Any issues found in steps 1 to 4 on page 60 suggest that there are problems with the network
2. Any issues found in steps 5 to 7 suggest that there are problems with the Tivoli Workload Scheduler configuration or installation

If this information does not provide the answer to the linking issue, call IBM Software Support for further assistance.

The commands used in steps 1 to 4 on page 60 are IP network management commands, information about which can be obtained in the Internet. The following technical note also provides useful information about their use: http://www.ibm.com/support/docview.wss?rs=0&uid=swg21156106

Losing a domain manager

A domain manager might need to be changed as the result of network linking problems or the failure of the domain manager workstation itself. It can be temporarily replaced by switching any full status agent in its domain to become the new domain manager, while the failed domain manager is repaired or replaced.

The steps for performing this activity are as described for the planned replacement of a domain manager; see *Tivoli Workload Scheduler: Administration Guide*.

Losing a master domain manager

If you lose a master domain manager, you have to perform all of the steps described in *Tivoli Workload Scheduler: Administration Guide* for the planned replacement of a master domain manager.

Other common network problems

The following problems could be encountered:

- "Submitted job is not running on agent with dynamic scheduling capabilities" on page 66
- "Job status of a submitted job is continually shown as running on agent with dynamic scheduling capabilities" on page 66

Using SSL, no connection between a fault-tolerant agent and its domain manager

In a network using SSL authentication, no connection can be established between a fault-tolerant agent and its domain manager. The standard lists of the two workstations display messages like the following:

- On the domain manager, mailman messages:

```
+ ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+ AWSBCV082I Workstation FTAHP, Message: AWSDEB009E Data
+ transmission is not possible because the connection is broken.
+ The following gives more details of the error: Error 0.
+ ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+ AWSBCV035W Mailman was unable to link to workstation: rsmith297;
+ the messages are written to the PO box.
+ ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

- On the fault-tolerant agent, writer messages:

```
/* ****************************************************************
/* AWSBCW003E Writer cannot connect to the remote mailman. The
/* following gives more details of the error: "
/* AWSDEB046E An error has occurred during the SSL handshaking. The
/* following gives more details of the error: error:140890B2:SSL
/* routines:SSL3_GET_CLIENT_CERTIFICATE:no certificate returned
/* ****************************************************************
/* AWSDEZ003E **ERROR**(cpu secs 0)
```

Cause and solution:

In the localopts file of either the domain manager or the fault-tolerant agent , the SSL port statement is set to 0.

Correct the problem by setting the SSL port number to the correct value in the localopts file. You then need to stop and restart netman on the workstation so that it can now listen on the correct port number.

After changing SSL mode, a workstation cannot link

You have changed the SSL mode between a workstation and its domain manager. However, you are unable to relink to the workstation from the domain manager.

Cause and solution:

The Symphony file and message files at the workstation must be deleted after a change of SSL mode, otherwise the data does not match. The files to delete are the following:

```
Symphony
Sinfonia
$HOME/*.msg
$HOME/pobox/*.msg
```

In a configuration with a firewall, the start and stop remote commands do not work

In a configuration with a firewall between the master domain manager and one or more domain managers, the **start** and **stop** commands from the master domain manager to the fault-tolerant agents in the domains do not work. This is often the case when an "rs final" ends and the impacted fault-tolerant agents are not linked.

Cause and solution:

The fault-tolerant agents belonging to these domains do not have the *behind firewall* attribute set to *on* in the Tivoli Workload Scheduler database. When there is a firewall between the master domain manager and other domains, **start** and **stop** commands must go through the Tivoli Workload Scheduler hierarchy. This parameter tells the master domain manager that the **stop** request must be sent to the domain manager which then sends it to the fault-tolerant agents in its domain.

Use either the Job Scheduling Console or the composer **cpuname** command to set to the *behind firewall* attribute *on* in the workstation definitions of these fault-tolerant agents.

The domain manager cannot link to a fault-tolerant agent

The domain manager cannot link to a fault-tolerant agent. The stdlist records the following messages:

```
+ ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+ AWSEDW020E: Error opening IPC
+ AWSEDW001I: Getting a new socket: 9
+ ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Cause and solution:

The fault-tolerant agent has two netman processes listening on the same port number. This is the case if you installed more than one Tivoli Workload Scheduler instance on the same workstation and failed to specify different netman port numbers.

Stop one of the two netman services and specify a unique port number using the `nm port` local option (`localopts` file).

Ensure that the workstation definition on the master domain manager is defined with the unique port number or it will not be able to connect.

Changes to the SSL keystore password prevent the application server from starting

You change the password to the SSL keystore on the application server, or you change the security settings using the WebSphere Application Server **changeSecuritySettings** tool. The application server does not start. The following message is found in the application server's trace file `trace.log` (the message is shown here on three lines to make it more readable):

```
JSAS0011E: [SSLConfiguration.validateSSLConfig] Java. exception
Exception = java.io.IOException:
Keystore was tampered with, or password was incorrect
```

This problem is discussed in "The application server does not start after changes to the SSL keystore password" on page 90.

Agents not linking to master domain manager after first JnextPlan on HP-UX

You have successfully installed the components of your network with the master domain manager on HP-UX. You perform all the necessary steps to create a plan and run your first **JnextPlan**, which appears to work correctly. The Symphony file is distributed to the agents but they cannot link to the master domain manager, even if you issue a specific **link** command for them. The **conman** error log shows that the agents cannot communicate with the master domain manager.

Cause and solution:

One possible cause for this problem is that while on HP-UX host names are normally limited to eight bytes, on some versions of this platform you can define larger host names. The problem occurs if you define the master domain manager's host name as more than eight bytes. When you install the master domain manager on this host a standard operating system routine obtains the host name from the operating system, but either truncates it to eight bytes before storing it in the database, or stores it as "unknown". When you install the agents, you supply the longer master domain manager host name. However, when the agents try to link to the master domain manager they cannot match the host name.

To resolve this problem, do the following:

1. Change the workstation definition of the master domain manager to the correct host name
2. Run **ResetPlan -scratch**
3. Run **JnextPlan**

The agents now link.

Fault-tolerant agents not linking to master domain manager

A fault-tolerant agent does not link to its master domain manager and any other link problem scenarios documented here do not apply.

Cause and solution:

The cause of this problem might not be easy to discover, but is almost certainly involved with a mismatch between the levels of the various files used on the fault-tolerant agent.

To resolve the problem, if all other attempts have failed, perform the following cleanup procedure. However, note that this procedure loses data (unless the fault-tolerant agent is not linking after a fresh installation), so should not be undertaken lightly.

Do the following:

1. Using **conman "unlink @;noask"** or the Tivoli Dynamic Workload Console, unlink the agent from the master domain manager
2. Stop Tivoli Workload Scheduler, in particular netman, as follows:
   a. **conman "stop;wait"**
   b. **conman "shut;wait"**

c. On Windows only; **shutdown**
d. Stop the SSM agent, as follows:
   - On Windows, stop the Windows service: Tivoli Workload Scheduler SSM Agent (for <*TWS_user*>).
   - On UNIX, run **stopmon**.

Note: If the **conman** commands do not work, try the following:

UNIX ps -ef |grep <TWS_user> & kill -9

Windows
<TWA_home>\TWS\unsupported\listproc & killproc

3. **Risk of data loss:** Removing the followoing indicated files can cause significant loss of data. Further, if jobs have run on the fault-tolerant agent for the current plan, without additional interaction, the fault-tolerant agent will rerun those jobs.

   Remove or rename the following files:

```
<TWS_home>\TWS\*.msg
              \Symphony
              \Sinfonia
              \Jobtable
              \pobox\*.msg
```

   Note: See Chapter 10, "Corrupt Symphony file recovery," on page 155 for additional options.
4. Start **netman** with **StartUp** run as the TWS_user
5. Issue a "**link**" command from the master domain manager to the fault-tolerant agent
6. Issue a **conman start** command on the fault-tolerant agent.

The IBM technical note describing this procedure also contains some advice about starting with a lossless version of this procedure (by omitting step 3) and then looping through the procedure in increasingly more-aggressive ways, with the intention of minimizing data loss. See http://www.ibm.com/support/docview.wss?uid=swg21296908

Agent with dynamic scheduling capabilities cannot be found from Tivoli Dynamic Workload Console

You correctly installed an agent that has dynamic scheduling capabilities but cannot see it from the Tivoli Dynamic Workload Console.

Cause and solution:

A possible cause for this problem might be that the dynamic workload broker hostname, **-tdwbhostname**, or the dynamic workload broker port, that is registered on the agent is not known in the network of the master domain manager because it is in a different DNS domain.

Edit the JobManager.ini configuration file. On Windows, It is located in <*TWA_home*>\ ita\JobManager.ini. On UNIX, it is located in bin/JobManager.ini. Edit the following parameter:

```
ResourceAdvisorUrl = https://<servername>:
                     31116/JobManagerRESTWeb/JobScheduler/resource
```

Submitted job is not running on agent with dynamic scheduling capabilities

From Tivoli Dynamic Workload Console, you can see an agent that has dynamic scheduling capabilities, but the submitted job appears as "No resources available" or is dispatched to other agents.

Cause and solution:

A possible cause might be that the local hostname of a registered dynamic workload broker server on the agent is not known in the network of the master domain manager because it is in a different DNS domain.

Edit the JobManager.ini configuration file. On Windows, it is located in <*TWA_home*>\ita\JobManager.ini. On UNIX, it is located in bin/JobManager.ini. Edit the following parameter:

```
FullyQualifiedHostname = <servername>
```

Job status of a submitted job is continually shown as running on agent with dynamic scheduling capabilities

From Tivoli Dynamic Workload Console, you can see an agent that has dynamic scheduling capabilities, but the job status of a submitted job is continually in the running state.

Cause and solution:

A possible cause might be that the master domain manager local hostname is not known in the network of the agent because it is in a different DNS domain.

Open the JobDispatcherConfig.properties file and edit the parameter JDURL=https://<*localhostname*>

Chapter 5. Troubleshooting common engine problems

This section details commonly occurring problems and their solutions in components and activities not already discussed in previous chapters. The problems are grouped according to their typology:

Other common problems are dealt with in other guides, or other chapters of this guide:

- For installation problems see the *Tivoli Workload Scheduler: Planning and Installation Guide*.
- For network problems see Chapter 4, "Troubleshooting networks," on page 59
- For problems with the fault-tolerant switch manager see Chapter 9, "Troubleshooting the fault-tolerant switch manager," on page 147.
- For problems with the Symphony file see Chapter 10, "Corrupt Symphony file recovery," on page 155.

Composer problems

The following problems could be encountered with composer:

Composer gives a dependency error with interdependent object definitions

You are running composer to add or modify a set of object definitions where one object is dependent on another in the same definition. An error is given for the dependency, even though the syntax of the definition is correct.

Cause and solution:

Composer validates objects in the order that they are presented in the command or the definition file. For example, you define two jobs, and the first-defined (*job_tom*) has a `follows` dependency on the second-defined (*job_harry*). The object validation tries to validate the `follows` dependency in *job_tom* but cannot find *job_harry* so gives an error and does not add the job to the database. However, it then reads the definition of *job_harry*, which is perfectly valid, and adds that to the database.

Similarly, this problem could arise if you define that a job needs a given resource or a job stream needs a given calendar, but you define the resource or calendar *after* defining the job or job stream that references them.

This problem applies to all composer commands that create or modify object definitions.

To resolve the problem, you can just simply repeat the operation. In the above example the following happens:

- The first job defined (*job_tom*) now finds the second job (*job_harry*) which was added to the database initially.
- You receive a "duplicate job" error for the second.

Alternatively, you can edit the object definition and retry the operation with just the object definition that gave the error initially.

To ensure that the problem does not reoccur, always remember to define objects in the order they are to be used. Define depending jobs and job streams before dependent ones. Define referred objects before referring objects.

Note: There is a special case of this error which impacts the use of the **validate** operation. Because **validate** does not add *any* job definitions to the database, correct or otherwise, *all* interdependent job definitions give an error.

In the example above, the problem would not have occurred when using add, new, create, or modify if the job definition of *job_harry* preceded that of *job_tom*. *job_harry* would have been added to the database, so the validation of *job_tom* would have been able to verify the existence of *job_harry*. Because the **validate** command does not add *job_harry* to the database, the validation of the `follows` dependency in *job_tom* fails.

There is no workaround for this problem when using **validate**. All you can do is to ensure that there are no interdependencies between objects in the object definition file.

The `display cpu=@` command does not work on UNIX

In UNIX, nothing happens when typing **display cpu=@** at the composer prompt.

Cause and solution:

The @ (atsign) key is set up as the "kill" character.

Type **stty -a** at the UNIX prompt to determine the setting of the @ key. If it is set as the "kill" character, then use the following command to change the setting to be "control/U" or something else:

```
stty kill ^U
```

where ^U is "control/U", not caret U.

The composer CLI does not start

You launch composer from the CLI but it does not start. Messages like the following might be given (the example is from UNIX):

```
Incorrectly built binary which accesses errno or h_errno
directly. Needs to be fixed.
bin/composer: error while loading shared libraries:
libstdc++-libc6.1-2.so.3: cannot open shared object file:
No such file or directory
```

Cause and solution:

This problem occurs when the environment variables for the CLI have not been set up correctly.

To set up the PATH and TWS_TISDIR variables, run one of the following scripts:

- **. ./***<TWA_home>***/TWS/tws_env.sh** for Bourne and Korn shells on UNIX
- **. ./***<TWA_home>***/TWS/tws_env.csh** for C shells on UNIX
- *<TWA_home>***\TWS\tws_env.cmd** on Windows

Composer gives the error "user is not authorized to access server"

You successfully launch composer but when you try to run a command, the following error is given:

```
user is not authorized to access server
```

Cause and solution:

This is a problem that is common to several CLI programs; see “Command line programs (like composer) give the error "user is not authorized to access server"” on page 104.

When using the composer add and replace commands, a job stream has synchronicity problems

The composer **add** and **replace** commands do not correctly validate the time zone used in the job stream definition at daylight savings; as a consequence, the following unexpected warning message is displayed:

```
AWSBIA148W  WARNING: UNTIL time occurs before AT time for
             <workstation>#<schedule>.
AWSBIA019E  For <workstation>#<schedule> Errors 0, warnings 1.
AWSBIA106W  The schedule definition has warnings.
AWSBIA015I  Schedule <workstation>#<schedule> added.
```

The same might happen for the `deadline` keyword.

Cause and solution:

The problem is related to the C-Runtime Library `date` and `time` functions that fail to calculate the correct time during the first week of daylight savings time.

To ensure the accuracy of scheduling times, for the time argument of the `at`, `until`, or `deadline` scheduling keywords, specify a different value than that of the start time for the Tivoli Workload Scheduler production period defined in the global options file. These values must differ from one another by plus or minus one hour.

JnextPlan problems

The following problems could be encountered with JnextPlan:

JnextPlan fails to start

JnextPlan fails to start.

Cause and solution:

This error might be a symptom that your Tivoli Workload Scheduler network requires additional tuning because of a problem with the sizing of the `pobox` files. The default size of the `pobox` files is 10MB. You might want to increase the size according to the following criteria:

- The role (master domain manager, domain manager, or fault-tolerant agent) of the workstation in the network. Higher hierarchical roles need larger pobox files due to the larger number of events they must handle (since the total number of events that a workstation receives is proportional to the number of its connections). For a domain manager, also the number of sub domains under its control make a difference.
- The average number of jobs in the plan.
- The I/O speed of the workstation (Tivoli Workload Scheduler is IO- dependent).

JnextPlan fails with the database message "The transaction log for the database is full."

You receive a message from **JnextPlan** which includes the following database message (the example is from DB2, but the Oracle message is very similar):

```
The transaction log for the database is full.
```

The **JnextPlan** message is probably the general database access error message AWSJDB801E.

Cause and solution:

The problem is probably caused by the number of job stream instances that **JnextPlan** needs to handle. The default database transaction log files cannot handle more than the transactions generated by a certain number of job stream instances. In the case of DB2 this number is 180 000; in the case of Oracle it depends on how you configured the database. If **JnextPlan** is generating this many instances, you need to change the log file creation parameters to ensure more log space is created. You might also need to increase the Java heap size on the application server. See "Scalability" in the *Tivoli Workload Scheduler: Administration Guide* for a full description of how to perform these activities.

JnextPlan fails with a Java out-of-memory error

You receive the following messages from **JnextPlan**:

```
AWSJCS011E An internal error has occurred.
The error is the following: "java.lang.OutOfMemoryError".
```

Cause and solution:

The problem is probably caused by the number of jobs that **JnextPlan** needs to handle. The default Java heap size in the application server cannot handle more than about 40 000 jobs. If **JnextPlan** is handling this many jobs, you need to increase the Java heap size. See "Scalability" in the *Tivoli Workload Scheduler: Administration Guide* for a full description of how to do this.

JnextPlan fails with the DB2 error like: nullDSRA0010E

JnextPlan has failed with the following messages:

```
AWSJPL705E An internal error has occurred. The planner is unable to create
the preproduction plan.

AWSBIS348E An internal error has occurred. MakePlan failed while running:
planman.

AWSBIS335E JnextPlan failed while running: tclsh84
```

The SystemOut.log has an error like this:

```
AWSJDB801E An internal error has been found while accessing the database.
The internal error message is: "nullDSRA0010E: SQL State = 57011, Error
Code = -912".
```

Cause and solution:

This indicates that the memory that DB2 allocates for its "*lock list*" is insufficient. To understand why the problem has occurred and resolve it, see the section in the *Tivoli Workload Scheduler: Administration Guide* about monitoring the "*lock list*" value among the DB2 administrative tasks.

JnextPlan fails with message AWSJPL017E

You receive the following message from **JnextPlan**:

```
AWSJPL017E The production plan cannot be created because a
previous action on the production plan did not complete successfully.
See the message help for more details.
```

Cause and solution:

The problem might be caused by a **JnextPlan** being launched before the previous **JnextPlan** has run the **SwitchPlan** command.

The situation might not resolve itself. To resolve it yourself, do the following:

1. Reset the plan by issuing the command **ResetPlan -scratch**
2. If the reset of the plan shows that the database is locked, run a **planman unlock** command.

JnextPlan is slow

You find that JnextPlan is unacceptable slow.

Cause and solution:

There are three possible causes for this problem:

Autotrace tracing too much

One possible cause is Autotrace. It could be that Autotrace is providing too much trace information. There are three possible solutions:

- Reduce the number of processes that Autotrace is monitoring. See "Changing the number of functions monitored" on page 55 for full details.
- Stop Autotrace while JnextPlan is running. To do this issue the following command before it starts:

  ```
  atctl off TWS all
  ```

  Issue the following command to switch Autotrace back on again:

  ```
  atctl on TWS all
  ```

  This can be automated within a script that launches JnextPlan.

Application server tracing too much

Another possible cause is that the application server tracing is set to high. See “Log and trace files for the application server” on page 14 for more details about the trace and how to reset it.

Database needs reorganizing

Another possible cause is that the database needs reorganizing. See "Reorganizing the database" in *Tivoli Workload Scheduler: Administration Guide* for a description of how and why you reorganize the database, logically and physically.

A remote workstation does not initialize after JnextPlan

After running JnextPlan you notice that a remote workstation does not immediately initialize. The following message is seen:

```
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+ AWSBCW037E  Writer cannot initialize this workstation because mailman
+ is still active.
+ ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+ AWSBCW039E Writer encountered an error opening the Mailbox.msg file.
+ The total cpu time used is as follows:  0
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Cause and solution:

If mailman is still running a process on the remote workstation, JnextPlan cannot download the Symphony file and initialize the next production period's activities. Instead, the domain manager issues a **stop** command to the workstation. The workstation reacts in the normal way to the **stop** command, completing those activities it must complete and stopping those activities it can stop.

After the interval determined in the `localopts` parameter *mm retrylink*, the domain manager tries again to initialize the workstation. When it finds that the **stop** command has been implemented, it starts to initialize the workstation, downloading the Symphony file and starting the workstation's activities.

A job remains in "exec" status after JnextPlan but is not running

After running **JnextPlan** you notice that a job has remained in exec" status, but is not being processed.

Cause and solution:

This error scenario is possible if a job completes its processing at a fault-tolerant agent just before **JnextPlan** is run. The detail of the circumstances in which the error occurs is as follows:

1. A job completes processing
2. The fault-tolerant agent marks the job as "succ" in its current Symphony file
3. The fault-tolerant agent prepares and sends a job status changed event (JS) and a job termination event (JT), informing the master domain manager of the successful end of job
4. At this point **JnextPlan** is started on the master domain manager

5. **JnextPlan** starts by unlinking its workstations, including the one that has just sent the JS and JT events. The message is thus not received, and waits in a message queue at an intermediate node in the network.
6. **JnextPlan** carries the job forward into the next Symphony file, and marks it as "exec", because the last information it had received from the workstation was the Launch Job Event (BL).
7. **JnextPlan** relinks the workstation
8. The fault-tolerant agent receives the new Symphony file and checks for jobs in the "exec" status.
9. It then correlates these jobs with running processes but does not make a match, so does not update the job status
10. The master domain manager receives the Completed Job Event that was waiting in the network and marks the carried forward job as "succ" and so does not send any further messages in respect of the job
11. Next time **JnextPlan** is run, the job will be treated as completed and will not figure in any further Symphony files, so the situation will be resolved. However, in the meantime, any dependent jobs will not have been run. If you are running **JnextPlan** with an extended frequency (for example once per month), this might be a serious problem.

There are two possible solutions:

Leave JnextPlan to resolve the problem
If there are no jobs dependent on this one, leave the situation to be resolved by the next **JnextPlan**.

Change the job status locally to "succ"
Change the job status as follows:

1. Check the job's stdlist file on the fault-tolerant agent to confirm that it did complete successfully.
2. Issue the following command on the fault-tolerant agent:

   ```
   conman "confirm <job>;succ"
   ```

To prevent the reoccurrence of this problem, take the following steps:

1. Edit the **JnextPlan** script
2. Locate the following command:

   ```
   conman "stop @!@;wait ;noask"
   ```
3. Replace this command with individual stop commands for each workstation (`conman "stop <workstation> ;wait ;noask"`) starting with the farthest distant nodes in the workstation and following with their parents, and so on, ending up with the master domain manager last. Thus, in a workstation at any level, a message placed in its forwarding queue either by its own job monitoring processes or by a communication from a lower level should have time to be forwarded at least to the level above before the workstation itself is closed down.
4. Save the modified **JnextPlan**.

A change in a resource quantity in the database is not also implemented in the plan after JnextPlan

You make changes to the number of available resources in the database, but the number of available resources in the plan does not change. The global option `enCFResourceQuantity` is set to *no*.

Cause and solution:

If the global option `enCFResourceQuantity` is set to *yes*, you would expect that any changes to the available quantity of a given resource in the database would not be implemented in the plan, provided there is at least one job or job stream instance using that resource in the extended plan.

Similarly, if the global option `enCFResourceQuantity` is set to *no* you might expect that the available resource quantity would change after **JnextPlan**. However, this is not always true, depending on the quantity of that resource being used by jobs and job stream instances currently in the plan:

- If the usage of the resource by jobs and job stream instances is *less than or equal to* the new total of available resources in the database, the available quantity of the resource is changed in the plan.
- If the usage of the resource by jobs and job stream instances is *greater than* the new total of available resources in the database, the available quantity of the resource is *not* changed in the plan.

To be sure to update the quantity of resources in the plan, make available at least as many instances of the resource as are required by the jobs and job stream instances in the plan.

See also the description of the `enCFResourceQuantity` option in the *Tivoli Workload Scheduler: User's Guide and Reference*.

On SLES8, after the second JnextPlan, an agent does not link

You have installed an agent on SLES8. The first **JnextPlan** works fine, but the second fails, with **conman** giving an error.

Cause and solution:

The problem is caused by a missing library on the agent workstation, called `libgcc_s.so.1`.

The **conman** process cannot run without this library, and **JnextPlan** uses **conman** to stop Tivoli Workload Scheduler processes that were started after the Symphony file arrived after the first **JnextPlan**. That is why **JnextPlan** did not fail the first time, because **JnextPlan** detected that processes were not running and did not need to use **conman** to stop them.

This is a library that is normally in `/lib`, but in this case is not. Look for it in other directories, such as `/usr/lib`. If you cannot locate it on your computer, contact IBM Software Support for assistance.

When you have located it, make a soft link to it from the `/lib` directory and rerun **JnextPlan**.

Conman problems

The following problems could be encountered when running **conman**:

- "Conman start does not work on UNIX systems running Autotrace"
- "On Windows, the message AWSDEQ024E is received"
- "Conman on a SLES8 agent fails because a library is missing" on page 77
- "Duplicate ad-hoc prompt number" on page 78
- "Submit job streams with a wildcard loses dependencies" on page 78

Conman start does not work on UNIX systems running Autotrace

On UNIX systems that support autotrace, the **conman start** command, run as *root* user, gives an error saying that it is not possible to load the atrc library.

Cause and solution:

This is a product limitation. Possible workarounds are:

- Use the StartUp utility instead of **conman start**
- Run **conman start** as the *TWS_user*

On Windows, the message AWSDEQ024E is received

When attempting to log in to **conman** on a Windows operating system, the following error is received:

```
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+ AWSDEQ024E Error owner is not of type user in TOKENUTILS.C;1178
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Cause and solution:

This problem can have a variety of causes related to users and permissions. Check the following on the server:

<*TWS_user*> **password**
: Make sure that the password that you supplied for the <*TWS_user*> user is correct, that the account is not locked out, and that the password has not expired.

Tokensrv service
: Ensure that the Tivoli Token Service (tokensrv) is started by the Tivoli Workload Scheduler administrative user (not the local system account). This must be verified in the properties of that service in the Services panel; see *Tivoli Workload Scheduler: Administration Guide* for details of how to access that panel and view the details of the user that "owns" the service.

  If the password to this user has changed on the workstation, check also that the password has been changed in the entry on the Services panel.

File ownerships
: Check that the following ownerships are correct:

- All `.exe` and `.dll` files in the <*TWA_home*>\TWS\bin directory are owned by the <*TWS_user*>
- All `.cmd` files are owned by "Administrator"

If necessary, alter the ownership of these files as follows:

1. Stop any active Tivoli Workload Scheduler processes.
2. Change to the <*TWA_home*>\TWS directory.
3. Issue the following commands:

```
setown -u <TWS_user> .\bin\*.exe
setown -u <TWS_user> .\bin\*.dll c:\win32app\maestro>
setown -u administrator .\bin\*.cmd
```

4. Issue a **StartUp** command on the affected server.
5. On the Tivoli Workload Scheduler master domain manager, launch **conman**.
6. Once conman is started, issue the following command sequence: **link @!@;noask**
7. Keep issuing the **sc** command to ensure that all the servers relink. A server is considered linked if the State shows "LTI JW"

Advanced user rights

Make sure that the <*TWS_user*> has the correct advanced user rights, as documented in the *Tivoli Workload Scheduler: Planning and Installation Guide*. These are as follows:

- Act as part of the operating system
- Adjust memory quotas for a process
- Log on as a batch job
- Log on as a service
- Log on locally
- Replace a process level token
- Impersonate a client after authentication right

Resolving the problem by reinstalling

If none of the above suggestions resolve the problem, you might need to reinstall Tivoli Workload Scheduler. However, it might happen that the uninstallation fails to completely remove all of the Registry keys from the previous installation. In this case, remove the registry keys following the procedure in the *Tivoli Workload Scheduler: Planning and Installation Guide*. Then make a fresh installation from the product DVD, subsequently reapplying the most recent fix pack, if any.

Conman on a SLES8 agent fails because a library is missing

You are running **conman** on an agent on Linux SLES8. A message is received indicating that **conman** cannot be run because the library `libgcc_s.so.1` is missing.

Cause and solution:

This is a library that is normally in `/lib`, but in this case is not. Look for it in other directories, such as `/usr/lib`. If you cannot locate it on your computer, contact IBM Software Support for assistance.

When you have located it, make a soft link to it from the `/lib` directory and rerun **JnextPlan**.

Duplicate ad-hoc prompt number

You issue a job or job stream that is dependent on an ad-hoc prompt, but conman cannot submit the job because the prompt number is duplicated.

Cause and solution:

On the master domain manager, prompts are created in the plan using a unique prompt number. This number is maintained in the file of the master domain manager. **JnextPlan** initially sets the prompt number to "1", and then increments it for each prompt that is to be included in the plan.

If you want to submit a job or job stream using an ad-hoc prompt on another Tivoli Workload Scheduler agent during the currency of a plan, the local **conman** looks in its own `runmsgno` file in its own `<TWA_home>/TWS/mozart/` directory, and uses the number it finds there. The value in the local file does not necessarily reflect the current value used in the `Symphony` file. For example, when the file is first created on an agent the run number is created as the highest run number used in the `Symphony` file at that time, plus *1000*. It is then incremented every time **conman** needs to assign a number to a prompt. Despite this interval of 1000, it is still possible for duplicates to occur.

To resolve the problem, edit the file and change the number. An example of the file contents is as follows:

```
         0       1236
```

The format is as follows:

- The 10-digit last Symphony run number, right-justified, blank filled. This should not be edited.
- A single blank
- The 10-digit last prompt number, right-justified, blank filled.

For example:

```
12345678901234567890
         0         98
```

When modifying the last prompt number, remember that the least significant digit must always be in character position 21. This means that if the current number is "98" and you want to modify it to display "2098" then you must replaces two spaces with the "20", and not just insert the two characters. For example:

```
12345678901234567890
         0       2098
```

Save the file and rerun the submit. No error should be given by **conman**.

Submit job streams with a wildcard loses dependencies

You issue a submit of interdependent job streams using a wildcard. In certain circumstances you lose the dependencies in an anomalous way.

Cause and solution::

To understand the cause, follow this example, in which the job streams are represented by A, B, C, and their instances are represented by 1, 2:

1. You have the following job streams and jobs in the Symphony file:

   ```
   A1
   B1 (A1,C1)
   C1
   ```

   where B1 depends on A1 and C1.
2. You submit all the jobs, using:

   ```
   sbs @
   ```

   The planner creates the following job stream instances:

   ```
   A2
   B2 (A2,C1)
   C2
   ```

   B2 now depends on A2 and C1. This is correct, because at the moment of submitting the B2 job stream C2 did not exist, so the highest instance available was C1.
3. The planner then asks you to confirm that you want to submit the instances:

   ```
   Do you want to submit A2?
   Do you want to submit B2?
   Do you want to submit C2?
   ```
4. Assume that you do not want to submit the job streams A2 and C2, yet, so you reply "No" to the first and last questions. In these circumstances you lose the dependency on A2, but not on C1. This behavior is correct and logical but could be seen by some as anomalous.

To correct the situation, stop the agent on the workstation where the job stream is running and cancel the job stream. Then determine the correct sequence of actions to perform to achieve your desired objective and submit the appropriate jobs.

Fault-tolerant agent problems

The following problems could be encountered with fault-tolerant agents.

- "A job fails in heavy workload conditions"
- "Batchman, and other processes fail on a fault-tolerant agent with the message AWSDEC002E" on page 80
- "Fault-tolerant agents unlink from mailman on a domain manager" on page 80

A job fails in heavy workload conditions

A job fails on a fault-tolerant agent where a large number of jobs are running concurrently and one of the following messages is logged:

- "TOS error: No space left on device."
- "TOS error: Interrupted system call."

Cause and solution:

This problem could indicate that one or more of the CCLog properties has been inadvertently set back to the default values applied in a prior version (which used to occasionally impact performance).

See "Generating engine log files with CCLog" on page 8 and check that the TWSCCLog.properties file contains the indicated default values for the properties twsHnd.logFile.className and twsloggers.className.

If the correct default values are being used, contact IBM Software Support to address this problem.

Batchman, and other processes fail on a fault-tolerant agent with the message AWSDEC002E

The batchman process fails together with all other processes that are running on the fault-tolerant agent, typically mailman and jobman (and JOBMON on Windows 2000). The following errors are recorded in the stdlist log of the fault-tolerant agent:

```
+ ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+ AWSBCV012E Mailman cannot read a message in a message file.
+ The following gives more details of the error:
+ AWSDEC002E  An internal error has occurred. The following UNIX
+ system error occurred on an events file: "9" at line = 2212
+ ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Cause and solution:

The cause is a corruption of the file `Mailbox.msg`, probably because the file is not large enough for the number of messages that needed to be written to it.

Consider if it seems likely that the problem is caused by the file overflowing:

- If you are sure that this is the cause, you can delete the corrupted message file.

  All events lost: Following this procedure means that all events in the corrupted message file are lost.

  Perform the following steps:

  1. Use the **evtsize** command to increase the `Mailbox.msg` file. Ensure that the file system has sufficient space to accommodate the larger file.
  2. Delete the corrupt message file.
  3. Restart Tivoli Workload Scheduler by issuing the conman **start** command on the fault-tolerant agent.
- If you do not think that this is the answer, or are not sure, contact IBM Software Support for assistance.

Fault-tolerant agents unlink from mailman on a domain manager

A message is received in the maestro log on the domain manager from mailman for each of the fault-tolerant agents to which it is connected. The messages are as follows:

```
MAILMAN:06:15/ + ++++++++++++++++++++++++++++++++++++++++++++++++
MAILMAN:06:15/ + WARNING: No incoming from <<workstation>>
                 - disconnecting. [2073.25]
MAILMAN:06:15/ + ++++++++++++++++++++++++++++++++++++++++++++++++
```

These messages usually occur in the 30 - 60 minutes immediately following JnextPlan.

Cause and solution:

This problem is normally caused by a false timeout in one of the mailman processes on the domain manager. During the initialization period immediately following JnextPlan, the "`*.msg`" files on the domain manager might become filled with a backlog of messages coming from fault-tolerant agents. While mailman is processing the messages for one fault-tolerant agent, messages from other fault-tolerant agents are kept waiting until the configured time interval for communications from a fault-tolerant agent is exceeded, at which point mailman unlinks them.

To correct the problem, increase the value of the *mm response* and *mm unlink* variables in the configuration file `~maestro/localopts`. These values must be increased together in small increments (60-300 seconds) until the time-outs no longer occur.

Problems on Windows

You could encounter the following problems running Tivoli Workload Scheduler on Windows.

- "Interactive jobs are not interactive using Terminal Services"
- "The Tivoli Workload Scheduler services fail to start after a restart of the workstation" on page 82
- "The Tivoli Workload Scheduler for user service (batchup) fails to start" on page 82
- "An error relating to impersonation level is received" on page 83

Interactive jobs are not interactive using Terminal Services

You want to run a job at a Windows fault-tolerant agent, launching the job remotely from another workstation. You want to use Windows Terminal Services to launch the job on the fault-tolerant agent, either with the Job Scheduling Console or from the command line. You set the "is interactive" flag to supply some run time data to the job, and indicate the application program that is to be run (for example, `notepad.exe`). However, when the job starts running, although everything seems correct, the application program window does not open on the Terminal Services screen. An investigation at the fault-tolerant agent shows that the application program is running on the fault-tolerant agent, but Terminal Services is not showing you the window.

Cause and solution:

The problem is a limitation of Terminal Services, and there is no known workaround. All "interactive jobs" must be run by a user at the fault-tolerant agent, and cannot be run remotely, using Terminal Services. Jobs that do not require user interaction are not impacted, and can be run from Terminal Services without any problems.

The Tivoli Workload Scheduler services fail to start after a restart of the workstation

On Windows, both the Tivoli Token service and the Tivoli Workload Scheduler for user service (batchup) fail to start after a restart of the workstation on which they are running.

Cause and solution:

The user under which these services start might have changed password.

If you believe this to be the case, follow the procedure described in *Tivoli Workload Scheduler: Administration Guide*.

The Tivoli Workload Scheduler for user service (batchup) fails to start

The Tivoli Workload Scheduler for <*TWS_user*> service (sometimes also called *batchup*) does not start when the other Tivoli Workload Scheduler processes (for example, mailman and batchman) start on workstations running Windows 2000 and 2003 Server. This problem occurs on a fault-tolerant agent, either after a conman start command or after a domain manager switch. The Tivoli Token service and netman services are unaffected.

This problem does not impact scheduling, but can result in misleading status data.

Cause and solution:

The problem is probably caused either because the <TWS_user> has changed password, or because the name of the service does not match that expected by Tivoli Workload Scheduler. This could be because a change in the configuration of the workstation has impacted the name of the service.

To resolve the problem temporarily, start the service manually using the Windows Services panel (under **Administrative Tools**. The service starts and runs correctly. However, the problem could reoccur unless you correct the root cause.

To resolve the problem permanently, take the following steps:

1. If the <*TWS_user*> has changed password, ensure that the service has been changed to reflect the new password, as described in *Tivoli Workload Scheduler: Administration Guide*.
2. Look at the Windows Event Viewer to see if the information there explains why the service did not start. Resolve any problem that you find.
3. If the reason given for the failure of the service to start is the following, this normally means that there is a mismatch between the name of the installed service, and the name of the service that the mailman process calls when it starts:

   ```
   System error code 1060:
           The specified service does not exist as an installed service
   ```

   The normal reason for this is that the user ID of the <*TWS_user*> has changed. The <*TWS_user*> cannot normally be changed by you, so this implies some change that has been imposed externally. A typical example of this is if you

have promoted the workstation from *member server* to *domain controller*. When this happens, the local <*TWS_user*> is converted automatically to a domain user, which means that the domain name is prefixed to the user ID, as follows: <*domain_name*>\<*TWS_user*>.

The problem occurs because of the way Tivoli Workload Scheduler installs the service. If the workstation is *not* a domain controller the installation names the service: tws_maestro_<*TWS_user*>. If the workstation *is* a domain controller the installation names the service: tws_maestro_<*domain_name*>_<*TWS_user*>.

When batchman starts up it discovers that the <*TWS_user*> is a domain user. Batchman tries to use the domain user service name to start the batchup service. The action fails because the service on the workstation has the local user service name.

To resolve this problem you must change the name of this service, and to do this you are recommended to uninstall the Tivoli Workload Scheduler instance and re-install it.

An alternative, but deprecated, method is to change the name of the service in the Windows registry.

Attention: Making changes to the Windows Registry can make the operating system unusable. You are strongly advised to back up the Registry before you start.

If you decide to use this method you must edit the following keys:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\tws_maestro_<TWS_user>
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\tws_maestro_<TWS_user>
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet002\Services\tws_maestro_<TWS_user>
```

They must be changed to the following:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\tws_maestro_<domain_name>_<TWS_user>
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\tws_maestro_<domain_name>_<TWS_user>
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet002\Services\tws_maestro_<domain_name>_<TWS_user>
```

If you have changed the name of the service in the registry, you must ensure that the logon is correct. Open the **Log On** tab of the service in the Windows Services panel and change the account name, if necessary, to <*domain_name*>\<*TWS_user*>. You must also enter the password and confirm it.

An error relating to impersonation level is received

On Windows, an error is received when you try to use any of the Tivoli Workload Scheduler commands (for example, **conman**, **composer**, **datecalc**). The error message is similar to the following:

```
AWSDEQ008E Error opening thread token ../../src/libs/tokenutils.c:1380
message = Either a required impersonation level was not provided, or the
provided impersonation level is invalid
```

Cause and solution:

This issue occurs when the user account that is used to run the Tivoli Workload Scheduler command line does not have the user right: "Impersonate a client after authentication". This is a new security setting that was first introduced in the following service packs:

Windows 2000	Service Pack 4
Windows XP	Service Pack 2
Windows 2003	All versions.

The upgrade does not grant this right to existing users.

For full details of this right, see the appropriate Windows publication.

To resolve this problem, grant the user right "Impersonate a client after authentication" to all users that need to run Tivoli Workload Scheduler commands on the workstation. To do this, follow these steps:

1. Select **Start ► Programs ► Administrative Tools ► Local Security Policy**
2. Expand **Local Policies**, and then click **User Rights Assignment**.
3. In the right pane, double-click **Impersonate a client after authentication**.
4. In the Local Security Policy Setting dialog box, click **Add**.
5. In the Select Users or Group dialog box, click the user account that you want to add, click **Add**, and then click **OK**.
6. Click **OK**.

Extended agent problems

The following problem could be encountered with extended agents:

The return code from an extended agent job is not recognized

You have a network including Tivoli Workload Scheduler versions 8.5, 8.4, 8.3, 8.2, or 8.2.1 and Tivoli Workload Scheduler for Applications, version 8.1.1. An extended agent job (submitted either through the Job Scheduling Console or **conman**), has given an unrecognized return code.

Cause and solution:

If Tivoli Workload Scheduler does not receive a return code from the extended agent job, it substitutes the return code with the exit code of the method. If this last is zero, the job has finished successfully. If it is not zero, contact IBM Software Support for an explanation of the exit code and a resolution of the problem.

Planner problems

The following problem could be encountered with the planner:

There is a mismatch between job stream instances in the Symphony file and the preproduction plan

You notice that there are job stream instances in the Symphony file that are not in the preproduction plan.

Cause and solution:

Job streams are automatically deleted from the preproduction plan when they are completed. However, it is possible to set the "carryStates" global option (using **optman**) so that job streams with jobs in the *SUCC* status are carried forward. In this case such job streams are carried forward to the new Symphony file when the plan is extended, but are deleted from the preproduction plan if the job streams have been successfully completed. This is not an error. These job streams can remain in the current plan (Symphony file) and can even be run again.

To resolve the situation for a given plan, use **conman** or the Job Scheduling Console to delete the job stream instances from the plan.

To prevent the problem reoccurring, consider why the "carryStates" global option is set so that job streams with jobs in the *SUCC* status are carried forward. If it has been set in error, or is no longer required, change the settings of the option (using **optman**) so that this no longer happens.

Planman deploy error when deploying a plug-in

When using the planman deploy command to deploy a plug-in, the deploy fails with the following error:

```
AWSJCS011E An internal error has occurred. The error is the following:
"ACTEX0019E The following errors
from the Java compiler cannot be parsed:
error: error reading <file_name>; Error opening zip file
<file_name>
```

Cause and solution:

The .jar file identified in the message is corrupt. Check and correct the format of the file before retrying the deploy.

An insufficient space error occurs while deploying rules

When using the planman deploy command with the -scratch option to deploy all non-draft rules, the following error occurs:

```
AWSJCS011E An internal error has occurred. The error is the following:
"ACTEX0023E The Active Correlation Technology compiler cannot
communicate with the external Java compiler.
java.io.IOException: Not enough space".
```

Cause and solution:

This error occurs when there is insufficient swap space (virtual memory) to perform the operation.

Create more swap space or wait until there are fewer active processes before retrying the operation.

UpdateStats fails if it runs more than two hours (message AWSJCO084E given)

When running the **UpdateStats** command in a large plan, if the job run time exceeds two hours, the job fails, with messages that include the following:

```
AWSJCO084E The user "UNAUTHENTICATED" is not authorized to work with the
           "planner" process.
```

Cause and solution:

This error occurs because the large number of jobs in the plan has caused the job run time to exceed two hours, which is the default timeout for the user credentials of the embedded WebSphere Application Server.

To increase the timeout so that the **UpdateStats** command has more time to run, perform the following:

1. Locate the following file:
   `<TWA_home>/eWAS/profiles/twaprofile/config/cells/DefaultNode/security.xml`
2. Locate the parameter: `authValidationConfig="system.LTPA" timeout="120"`
3. Edit the timeout value from 120 minutes to a value you think will be sufficient.
4. Stop and restart the embedded WebSphere Application Server, using the conman **stopappserver** and **startappserver** commands (or, in the latter case, the **StartUp** command).

The planman showinfo command displays inconsistent times

The plan time displayed by the `planman showinfo` command might be incongruent with the time set in the operating system of the workstation. For example, the time zone set for the workstation is `GMT+2` but `planman showinfo` displays plan times according to the `GMT+1` time zone.

Cause and solution:

This situation arises when the WebSphere Application Server Java virtual machine does not recognize the time zone set on the operating system.

As a workaround for this problem, set the time zone defined in the `server.xml` file equal to the time zone defined for the workstation in the Tivoli Workload Scheduler database. Proceed as follows:

1. Stop WebSphere Application Server
2. Create a backup copy of the following file:
   `<TWA_home>/eWAS/profiles/twaprofile/config/cells/`
   `DefaultNode/nodes/DefaultNode/servers/server<n>.xml`
3. Open the original file with a text or XML editor
4. Find the `genericJvmArguments` string and add:
   `genericJvmArguments="-Duser.timezone=time_zone"`

   where *time_zone* is the time zone defined for the workstation in the Tivoli Workload Scheduler database.
5. Restart WebSphere Application Server

Problems with DB2

The following problems could be encountered with DB2:

- "Timeout occurs with DB2"
- "JnextPlan fails with the DB2 message "The transaction log for the database is full.""
- "The DB2 UpdateStats job fails after 2 hours"
- "DB2 might lock while making schedule changes" on page 88

Timeout occurs with DB2

You are trying to edit an object, but after a delay an error is given by DB2 referring to a timeout, similar to the following:

```
AWSJDB803E
An internal deadlock or timeout error has occurred while processing a
database transaction. The internal error message is:
"The current transaction has been rolled back because of a deadlock or timeout.
Reason code "68".
```

Cause and solution:

In this case the object you are trying to access is locked by another user, or by you in another session, but the lock has not been detected by the application. So the application waits to get access until it is interrupted by the DB2 timeout.

By default, both DB2 and WebSphere Application Server have the same length timeout, but as the WebSphere Application Server action starts before the DB2 action, it is normally the WebSphere Application Server timeout that is logged:

```
AWSJCO005E WebSphere Application Server has given the following error:
CORBA NO_RESPONSE 0x4942fb01 Maybe; nested exception is:
     org.omg.CORBA.NO_RESPONSE:
     Request 1685 timed out vmcid:
     IBM minor code: B01 completed: Maybe.
```

To resolve the problem, check if the object in question is locked. If it is, take the appropriate action to unlock it, working with the user who locked it. If it is not locked retry the operation. If the problem persists contact IBM Software Support for assistance.

JnextPlan fails with the DB2 message "The transaction log for the database is full."

You receive a message from **JnextPlan** which includes the following DB2 message:

```
The transaction log for the database is full.
```

The **JnextPlan** message is probably the general database access error message AWSJDB801E.

Cause and solution:

This scenario is described in "JnextPlan fails with the database message "The transaction log for the database is full."" on page 71.

The DB2 UpdateStats job fails after 2 hours

You are running the DB2 **UpdateStats** job, but after 2 hours it fails. The log contains messages similar to the following:

```
[2/20/08 8:22:11:947 CET] 0000001e ServiceLogger I
     com.ibm.ws.ffdc.IncidentStreamImpl initialize FFDC0009I:
     FFDC opened incident stream file /opt/ibm/TWA0/eWAS/profiles/
          twaprofile/logs/ffdc/server1_78387838_08.02.20_08.22.11_0.txt
[2/20/08 8:22:11:957 CET] 0000001e ServiceLogger I
     com.ibm.ws.ffdc.IncidentStreamImpl resetIncidentStream FFDC0010I:
     FFDC closed incident stream file /opt/ibm/TWA0/eWAS/profiles/
          twaprofile/logs/ffdc/server1_78387838_08.02.20_08.22.11_0.txt
[2/20/08 8:22:11:999 CET] 0000001e ConnException E
     com.ibm.tws.conn.exception.ConnSecurityException
          ConnException(String currentMessageID, Object[] currentArgs)
     AWSJCO084E The user "UNAUTHENTICATED" is not authorized to work with
                the "planner" process. UNAUTHENTICATED
[2/20/08 8:22:12:004 CET] 0000001e ConnException E
     com.ibm.tws.conn.exception.ConnException
          ConnException(TWSException e)
     AWSJCO084E The user "UNAUTHENTICATED" is not authorized to work with
                the "planner" process.
[2/20/08 8:22:12:088 CET] 0000001e ExceptionHelp E
     com.ibm.tws.cli.exception.ExceptionHelper
          handleException(Throwable e, String commandName,
               TWSServletResponse response)
     AWSJCL054E The command "LOGREPORT" has failed, for the following reason:
          "AWSJCO084E The user "UNAUTHENTICATED" is not authorized to work with
          the "planner" process.".
     LOGREPORT AWSJCO084E The user "UNAUTHENTICATED" is not authorized to work
          with the "planner" process.
[2/20/08 8:22:12:091 CET] 0000001e ThreadMonitor W
     WSVR0606W: Thread "WebContainer : 2" (0000001e) was previously reported
          to be hung but has completed.  It was active for approximately 7200340
          milliseconds.  There is/are 0 thread(s) in total in the server that
          still may be hung.
```

Cause and solution:

The problem is with the WebSphere Application Server which has a default authentication timeout of 2 hours. The **UpdateStats** job runs without any interrupt that would allow the WebSphere Application Server to reset its timeout.

To resolve the problem, reset the timeout as follows:

1. Edit the following file with a text editor:
   `<TWA_home>/eWAS/profiles/twaprofile/config/cells/DefaultNode/security.xml`
2. Locate the key: `authValidationConfig="system.LTPA" timeout="120"`
3. Change the value of the timeout to an appropriately higher figure (the log of **UpdateStats** shows you how much progress the job had made when it stopped; it should be possible to extrapolate from that how much additional time is required).
4. Save the file.
5. Stop and restart the application server using the **stopappserver** and **startappserver** commands.
6. Rerun **UpdateStats**.

DB2 might lock while making schedule changes

Multiple concurrent changes (modify, delete or create) to job streams or domains might cause a logical deadlock between one or more database transactions. This is a remote but possible problem you might encounter.

This deadlock might take place even if the objects being worked on are different (for example, different job streams).

The problem affects database elements (rows or tables), not Tivoli Workload Scheduler objects, so it is unrelated with the `Locked By` property of Tivoli Workload Scheduler objects.

The same problem might arise when making concurrent changes for plan generation.

When the deadlock occurs, DB2 rollbacks one of the deadlocking threads and the following error is logged in the `SystemOut.log` of WebSphere Application Server:

```
AWSJDB803E An internal deadlock or timeout error has occurred
while processing a database transaction. The internal error
message is: "The current transaction has been rolled back
because of a deadlock or timeout. Reason code "2"."
```

In general, this type of error is timing-dependent, and the transactions involved must overlap in very specific conditions to generate a deadlock. However it might easily occur during plan generation (either forecast, trial, or current), when the plan includes many objects and DB2 must automatically escalate locks from row to table level, as the number of locked objects exceeds the current maximum limit.

You can mitigate the error by increasing the maximum number of locks that DB2 can hold. Refer to the DB2 Information Center to learn more about the DB2 lock escalation mechanism and to find how to increase the maximum number of concurrent locks.

In the above scenarios, if an interactive user session is rolled back, the user gets an error message but is allowed to repeat the task. Instead, if a script session is rolled back (for example, a script that generates a forecast plan or updates a job stream definition), the script ends in failure.

Problems with Oracle

The following problems could be encountered with Oracle:

- “JnextPlan fails with the database message "The transaction log for the database is full."”
- “You cannot do Oracle maintenance on UNIX after installation”

JnextPlan fails with the database message "The transaction log for the database is full."

You receive a message from **JnextPlan** which includes a database message similar to the following:

```
The transaction log for the database is full.
```

The **JnextPlan** message is probably the general database access error message AWSJDB801E.

Cause and solution:

This scenario is described in “JnextPlan fails with the database message "The transaction log for the database is full."” on page 71.

You cannot do Oracle maintenance on UNIX after installation

You have installed Tivoli Workload Scheduler, creating the installation directory with the default root user permission. When you switch to the Oracle

administration user and try and use the Oracle tools, you encounter access problems.

Cause and solution:

The problem could be that the Oracle administration user does not have "read" permission for the entire path of the Tivoli Workload Scheduler installation directory. For example, if you have created the Tivoli Workload Scheduler installation directory as `/opt/myProducts/TWS`, the Oracle administration user must have "read" permission for `/opt` and `/myProducts`, as well as `/TWS`.

Give the Oracle administration user read permission for the full path of the Tivoli Workload Scheduler installation directory.

Application server problems

The following problems might occur:

- "Timeout occurs with the application server"
- "The application server does not start after changes to the SSL keystore password"

The application server does not start after changes to the SSL keystore password

You change the password to the SSL keystore on the application server, or you change the security settings using the WebSphere Application Server **changeSecuritySettings** tool. The application server does not start. The following message is found in the application server's trace file `trace.log` (the message is shown here on three lines to make it more readable):

```
JSAS0011E: [SSLConfiguration.validateSSLConfig] Java. exception
Exception = java.io.IOException:
Keystore was tampered with, or password was incorrect
```

Cause and solution:

The certificate has not been reloaded or regenerated. Any change to the keystore password on the server or connector requires the SSL certificate to be reloaded or regenerated to work correctly.

Reload or regenerate the certificate and restart the application server.

To regenerate the certificate issue this command:

```
openssl genrsa -des3 -passout pass:<your_password> -out client.key 1024
```

If you do not want to supply the password openly in the command, omit it, and you will be prompted for it.

Timeout occurs with the application server

You are trying to edit an object, but after a delay an error is given by the WebSphere Application Server referring to a timeout, similar to the following:

```
AWSJCO005E WebSphere Application Server has given the following error:
CORBA NO_RESPONSE 0x4942fb01 Maybe; nested exception is:
     org.omg.CORBA.NO_RESPONSE:
     Request 1685 timed out vmcid:
     IBM minor code: B01 completed: Maybe.
```

Cause and solution:

In this case the object you are trying to access is locked from outside Tivoli Workload Scheduler, maybe by the database administrator or an automatic database function. So the application waits to get access until it is interrupted by the application server timeout.

DB2 By default, both DB2 and WebSphere Application Server have the same length timeout, but as the WebSphere Application Server action starts before the DB2 action, it is normally the WebSphere Application Server timeout that is logged.

If one or both of the timeouts have been modified from the default values, and the DB2 timeout is now shorter, the following message is given:

```
AWSJDB803E
An internal deadlock or timeout error has occurred while processing a
database transaction. The internal error message is:
"The current transaction has been rolled back because of a
deadlock or timeout.
Reason code "68".
```

Oracle There is no corresponding timeout on Oracle, so the Tivoli Dynamic Workload Console hangs.

To resolve the problem, get the database administrator to check if the object in question is locked outside Tivoli Workload Scheduler. If it is, take the appropriate action to unlock it, if necessary asking the database administrator to force unlock the object.

If the object is not locked outside Tivoli Workload Scheduler, retry the operation. If the problem persists contact IBM Software Support for assistance.

Event management problems

This section describes problems that might occur with processing of events. The topics are as follows:

- "Preventing problems with disk usage, process status, and mailbox usage" on page 101

Troubleshooting an event rule that does not trigger the required action

You have created an event rule but the required action is not triggered when the event condition is encountered.

Cause and solution:

The cause and subsequent solution might be any of a number of things. Use the following check list and procedures to determine what has happened and resolve the problem. The check list uses a test event which has the following characteristics:

```
<eventRule name="TEST1" ruleType="filter" isDraft="no">
  <description>A Rule that checks the sequence of events</description>
 <eventCondition name="fileCreated1" eventProvider="FileMonitor"
  eventType="FileCreated">
   <scope>
    C:\TEMP\FILE5.TXT ON CPU_MASTER
   </scope>
   <filteringPredicate>
    <attributeFilter name="FileName" operator="eq">
     <value>c:\temp\file5.txt</value>
    </attributeFilter>
    <attributeFilter name="Workstation" operator="eq">
     <value>CPU_MASTER</value>
    </attributeFilter>
    <attributeFilter name="SampleInterval" operator="eq">
     <value>60</value>
    </attributeFilter>
   </filteringPredicate>
  </eventCondition>
  <action actionProvider="TWSAction" actionType="sbj" responseType="onDetection">
   <scope>
    SBJ CPU_MASTER#JOB1 INTO CPU_MASTER#JOBS
   </scope>
   <parameter name="JobUseUniqueAlias">
    <value>true</value>
   </parameter>
   <parameter name="JobDefinitionWorkstationName">
    <value>CPU_MASTER</value>
   </parameter>
   <parameter name="JobDefinitionName">
    <value>JOB1</value>
   </parameter>
  </action>
 </eventRule>
```

The check list is as follows:

Step 1: Is event management enabled?

Check if the event management feature is enabled (at installation it is enabled by default):

1. Run the following command:

   ```
   optman ls
   ```

   and look for the following entry:

   ```
   enEventDrivenWorkloadAutomation / ed = YES
   ```

If the value is "YES", go to Step 2.

2. **Action:** If the property is set to *NO*, run the command:

   ```
   optman chg ed=YES
   ```

3. To effect the change, run:

   ```
   JnextPlan –for 0000
   ```

   Check that the event rule is now being processed correctly. If not, go to Step 2.

Step 2: Is the workstation enabled for event processing?

Check that the workstation is enabled for event processing. By default the master domain manager and backup master domain manager are enabled for event processing, but the default value might have been changed. Do as follows:

1. View the `localopts` file on the master domain manager with a text editor or viewer, and check for the following entry:

   ```
   can be event processor = yes
   ```

   If the value is "yes", go to Step 3.

2. **Action:** If the value is "no", set it to "yes". Save the `localopts` file and stop and start Tivoli Workload Scheduler. Check that the event rule is now being processed correctly. If not, go to Step 3.

Step 3: Is the event processor installed, up and running, and correctly configured?

1. Start **conman**
2. Issue the **showcpus** command:

   ```
   %sc
   ```

   The output should be similar to the following:

```
CPUID        RUN NODE        LIMIT FENCE DATE     TIME  STATE       METHOD   DOMAIN
CPU_MASTER    11 *WNT  MASTER    0     0 09/03/07 09:51    I JW MDEA         MASTERDM
FTA1          11  WNT  FTA       0     0                LT                   MASTERDM
```

3. Check the STATE field for the presence of an M, a D, and an E (upper-case) (in the example, the STATE field has a value of *I JW MDEA*, and the *MDE* is highlighted). If *all* are present, the event processor is installed, up and running, and correctly configured; go to Step 8.
4. **Actions:** If one or more of M, D, and E are not present, perform one or more of the following actions until they are all present:

   The STATE field has neither an upper-case E nor a lower-case e

   If there is neither an upper-case E nor a lower-case e, the event processor is not installed. The event processor is installed by default on the master domain manager and backup master domain manager. If you are working on either, then the installation did not complete correctly. Collect the log files in the *<TWA_home>*`/TWS/stdlist` directory and contact IBM Software Support for assistance.

   The STATE field has a lower-case e

   If the STATE field has a lower case e, the event processor is installed but not running. Start the event processor using the

conman startevtproc command, or the Dynamic Workload Console. If you use **conman**, for example, you will see the following output:

```
%startevtproc
AWSJCL528I The event processor has been started successfully.
```

The STATE field has no M

If the STATE field has no M, **monman** is not running. Start **monman** using the **conman startmon** command. You will see the following output:

```
%startmon
AWSBHU470I A startmon command was issued for CPU_MASTER.
```

The STATE field has no D

If the STATE field has no D, the current monitoring package configuration is not deployed. Go to step 4.

5. Rerun the **showcpus** command.
6. When the M, D, and E are all present, check that the event rule is now being processed correctly. If not, go to Step 8.

Step 4: Has the rule been added to the monitoring configuration on the workstation?

1. Check if the rule is present in the workstation monitoring configuration by running the **conman showcpus** command with the **;getmon** argument:

```
%sc ;getmon
Monitoring configuration for CPU_MASTER:
*****************************************
***  Package date : 2008/09/03 07:48 GMT ***
*****************************************

TEST1::FileMonitor#FileCreated:C:\TEMP\FILE5.TXT ON CPU_MASTER;
TEST1::TWSObjectsMonitor#JobSubmit:* # * . TEST*;
```

   If the rule is present, go to Step 6.

2. **Action:** If the configuration does not contain the expected rule, go to step 5.

Step 5: Is the rule active

If the configuration does not contain the expected rule, check if it is active.

1. Check the rule status, using the **composer list** command or the Tivoli Dynamic Workload Console. If you use composer, for example, you will see output similar to the following:

```
-list er=@

Event Rule Name                Type       Draft  Status          Updated On  Locked By
-----------------------------  ---------  -----  --------------  ----------  ----------------
TEST1                          filter       N     active           09/03/2008  -
```

   If the rule is in *active* status go to Step 6.

2. **Action:** If the rule is in *error* status, activate the Tivoli Workload Scheduler trace, collect the log files in the <*TWA_home*>/TWS/stdlist directory and contact IBM Software Support for assistance.

Step 6: Has the new monitoring configuration been deployed to the workstation?

If the rule is active, check if the new monitoring configuration has been deployed to the workstation.

1. The deployment of a new monitoring configuration can be checked in either of these ways:
   - Check in the <*TWA_home*>/TWS/monconf if the configuration is present
   - Check in the SystemOut file in <*TWA_home*>/eWAS/profiles/twaprofile/logs/server<*n*>. Look for the message:

```
[9/3/07 9:50:00:796 CEST] 00000020 sendEventReadyConfiguration(wsInPlanIds, zipsToDeploy)
                          AWSDPM001I The workstation "CPU_MASTER" has been notified about
                          a new available configuration.
```

   If the message is present for the workstation in question after the time when the rule was made available for deployment, then the new configuration has been deployed.

   If the configuration has been deployed, go to Step 7.

2. **Action:** If the configuration has not been deployed, deploy it with the **conman deploy** command:

```
%deploy
AWSBHU470I A deployconf command was issued for MASTER_CPU.
```

   Check that the event rule is now being processed correctly. If not, go to Step 7.

Step 7: Has the deploy of the new monitoring configuration worked correctly?

If the new monitoring configuration has been deployed, check that the deployment was successful:

1. Check in the <*TWA_home*>/TWS/stdlist/traces/<*date*>_TWSMERGE.log, and look for the most recent occurrence of these 2 messages:

```
09:51:57 03.09.2008|MONMAN:INFO:=== DEPLOY ===> CPU_MASTER has been notified
                               of the availability of the new monitoring configuration.
09:51:57 03.09.2008|MONMAN:INFO:=== DEPLOY ===> The zip file d:\TWS\twsuser\monconf\deployconf.zip
                               has been successfully downloaded.
```

   If you find these messages, referring to the workstation in question, and occurring after the time when the rule was deployed, then the rule has been successfully deployed to the workstation: go to Step 8.

2. **Actions:** If you find messages that indicate an error, follow one of these actions:

   Message indicates that the server could not be contacted or that the action has been resubmitted by monman

   The message you find is either of the following:

```
=== DEPLOY ===> ERROR contacting the server for receiving the zip file (rc=8)

=== DEPLOY ===> The deploy action has been automatically resubmitted by monman.
```

   The application server could be down. Either wait for 5 minutes, or follow the instructions about how to use **appserverman** (see *Tivoli Workload Scheduler: Administration Guide*) to determine if the application server is down, and if it is being restarted automatically, or needs to be restarted manually.

If you need to change any aspect of the application server configuration, run **JnextPlan –for 0000**.

When you are certain that the application server is up, retry Step 7.

Message indicates a problem with decoding or unzipping the zip

The message you find is either of the following:

```
=== DEPLOY ===> ERROR decoding the zip file temporarily downloaded in
                                                  <TWA_home>/TWS/monconf

=== DEPLOY ===> ERROR unzipping the zip file <file_name>
```

Collect the log files and contact IBM Software Support for assistance.

Step 8: Is the SSM agent running (for rules with `FileMonitor` plug-in-related events only)?

1. If the rule has an event that uses the `FileMonitor` plug-in, check that the SSM Agent is running. Check in the log that when the **conman startmon** command was run (either when you ran it manually or when Tivoli Workload Scheduler started.
2. Then look ahead in the log for the following message:

   ```
   11:13:56 03.09.2008|MONMAN:INFO:SSM Agent service successfully started
   ```

   If it is present, or the rule does not use the `FileMonitor` plug-in, go to Step 5.
3. **Action:** If the SSM Agent message is not present, collect the log files in the *<TWA_home>*`/TWS/stdlist` directory and the *<TWA_home>*`/TWS/ssm/` directory and contact IBM Software Support for assistance.

Step 9: Have the events been received?

You know the rule has been deployed, but now you need to know if the event or events have been received.

1. Check in the `SystemOut` of the server to see if the event has been received. The output is different, depending on the type of event:

   FileMonitorPlugIn event

   a. This is the output of a FileMonitorPlugIn event:

```
[9/3/07 9:55:05:078 CEST] 00000035 EventProcessor A com.ibm.tws.event.EventProcessorManager
                          processEvent(IEvent)
                          AWSEVP001I The following event has been received:
                          event type = "FILECREATED"; event provider = "FileMonitor";
                          event scope = "c:\temp\file5.txt on CPU_MASTER".
                          FILECREATED FileMonitor c:\temp\file5.txt on CPU_MASTER
```

   If the event has been received, go to Step 10.

   b. If the event has not been received check if it has been created by looking in the `traps.log` for the message that indicates that the event has been created:

```
.1.3.6.1.4.1.1977.47.1.1.4.25 OCTET STRING FileCreatedEvent event
```

   c. **Action:** Whether the event has or has not been created, collect the information in the *<TWA_home>*`/TWS/ssm` directory and contact IBM Software Support for assistance.

TWSObjectMonitorPlugIn event

a. This is the output of a TWSObjectMonitorPlugIn event:

```
[9/3/07 12:28:38:843 CEST] 00000042 EventProcesso A com.ibm.tws.event.EventProcessorManager
                           processEvent(IEvent)
                           AWSEVP001I The following event has been received: event type = "JOBSUBMIT";
                           event provider = ""TWSObjectsMonitor""; event scope = "CPU_MASTER # JOBS .
                           (CPU_MASTER #) TEST". JOBSUBMIT "TWSObjectsMonitor" CPU_MASTER # JOBS .
                           (CPU_MASTER #) TEST
```

b. **Action:** If the event has not been received, collect the log data and contact IBM Software Support for assistance.

c. If the TWSObjectMonitorPlugIn event has been received, check in the same log that the EIF event has been sent. This is the output of an EIF event:

```
12:27:18 03.09.2008|MONMAN:INFO:Sending EIF Event:
                   "JobSubmit;
                    TimeStamp="2008-09-03T12:26:00Z/";
                    EventProvider="TWSObjectsMonitor";
                    HostName="CPU_MASTER";
                    IPAddress="9.71.147.38";
                    PlanNumber="11";
                    Workstation="CPU_MASTER";
                    JobStreamWorkstation="CPU_MASTER";
                    JobStreamId="JOBS";
                    JobStreamName="JOBS";
                    JobStreamSchedTime="2008-09-03T12:26:00";
                    JobName="TEST";
                    Priority="10";
                    Monitored="false";
                    EstimatedDuration="0";
                    ActualDuration="0";
                    Status="Waiting";
                    InternalStatus="ADD";
                    Login="twsuser";END
```

d. If the EIF event has been sent, it might be cached in the *<TWA_home>*/TWS/EIF directory.

e. If the event is found there, check the communication with the agent and the server. If no communication problem is present wait until the event is sent.

f. The event might also be cached in the machine where the event processor is located. Check this in the *<TWA_home>*/eWAS/profiles/twaprofile/temp/TWS/EIFListener. If the event is found there, check the communication with the agent and the server. If no communication problem is present wait until the event is sent.

2. **Action:** If the problem persists, collect the log data and contact IBM Software Support for assistance.

Step 10: Has the rule been performed?

You now know that the event has been received, but that the action has apparently not been performed.

1. Check in the SystemOut of the server to see if the rules have been performed. Look for messages like these:

```
[9/3/07 9:55:05:578 CEST] 00000035 ActionHelper  A com.ibm.tws.event.plugin.action.ActionHelper
                          invokeAction(ActionContext,Map,EventRuleHeader)
                          AWSAHL004I The rule "TEST1" has been triggered. TEST1
[9/3/07 9:55:05:625 CEST] 00000036 ActionHelper  A com.ibm.tws.event.plugin.action.ActionHelper
                          AsynchAction::run()
                          AWSAHL002I The action "sbj" for the plug-in "TWSAction" has been started.
```

```
                        sbj TWSAction
[9/3/07 9:55:06:296 CEST] 00000036 ActionHelper  A com.ibm.tws.event.plugin.action.ActionHelper
                        AsynchAction::run()
                        AWSAHL003I The action "sbj" for the plug-in "TWSAction" has completed.
                        sbj TWSAction
```

If the rule has been triggered and the action completed, go to step 11.

2. **Action:** If the action has not been completed collect the log data and contact IBM Software Support for assistance.

Step 11: Is the problem in the visualization of the event?

Action: If the event has been received, but you cannot see it, there might be a problem with the console you are using to view the event. For Tivoli Dynamic Workload Console troubleshooting see Chapter 7, "Troubleshooting Tivoli Dynamic Workload Console problems," on page 121. To troubleshoot the Job Scheduling Console, see the *Tivoli Workload Scheduler: Job Scheduling Console User's Guide*.

Actions involving the automatic sending of an e-mail fail

An event rule is created, including as the required action the sending of an e-mail. When the event occurs, the action fails with the following message:

```
AWSMSP104E  The mail "<mailID>" has not been successfully
            delivered to "<recipient>".
Reason: "Sending failed;
nested exception is:
?????class javax.mail.MessagingException: 553 5.5.4 <TWS>...
Domain name required for sender address TWS
```

Cause and solution:

The mail send action failed because the domain name of the SMTP server was not defined in the mail sender name global option: `mailSenderName` (`ms`).

Use the **optman** command to specify the correct mail sender name including the domain. For example, if the mail sender name is tws@alpha.ibm.com, issue the following command:

optman chg ms tws@alpha.ibm.com

An event is lost

You have sent a large number of events to the event processor. When you check the event queue you find that the most recent event or events are missing.

Cause and solution:

The event queue is not big enough. 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.

You cannot recover the event that has been overwritten, but you can increase the size of the queue to ensure the problem does not recur. Follow the instructions in "Managing the event queue" in *Tivoli Workload Scheduler: Administration Guide*.

Event rules not deployed after switching event processor

You have switched the event processor, but new or amended rules have not been deployed (the event states of the workstations that were affected by the new or amended rules do not show "D" indicating that the rules are not up-to-date, and the **getmon** command shows the old rules).

Cause and solution:

The probable cause is that you made some changes to the rules before running the **switcheventprocessor** command, and these rules were not deployed (for whatever reason) before the switch.

To remediate the situation, run the command **conman deployconf <***workstation_name***>**, for each affected workstation, and the rule changes will be deployed.

To avoid that this problem reoccurs, run **planman** with the *deploy* action before running **switcheventprocessor**.

Event *LogMessageWritten* is not triggered

You are monitoring a log file for a specific log message, using the *LogMessageWritten* event. The message is written to the file but the event is not triggered.

Cause and solution:

The SSM agent monitors the log file. It sends an event when a new message is written to the log file that matches the string in the event rule. However, there is a limitation. It cannot detect the very latest message to be written to the file, but only messages prior to the latest. Thus, when message line "n" is written containing the string that the event rule is configured to search for, the agent does not detect that a message has been written, because the message is the last one in the file. When any other message line is written, if or not it contains the monitored string, the agent is now able to read the message line containing the string it is monitoring, and sends an event for it.

There is no workaround to resolve this problem. However, it should be noted that in a typical log file, messages are being written by one or other processes frequently, perhaps every few seconds, and the writing of a subsequent message line will trigger the event in question. If you have log files where few messages are written, you might want to attempt to write a dummy blank message after every "real" message, in order to ensure that the "real" message is never the last in the file for any length of time.

Deploy (D) flag not set after ResetPlan command used

The deploy (D) flag is not set on workstations after the **ResetPlan** command is used.

Cause and solution:

This is not a problem that affects the processing of events but just the visualization of the flag which indicates that the event configuration file has been received at the workstation.

You can choose to do nothing, because the situation will be normalized the next time that the event processor sends an event configuration file to the workstation.

Alternatively, if you want to take a positive action to resolve the problem, do the following:

- Create a dummy event rule that applies only to the affected workstations
- Perform a **planman deploy** to send the configuration file
- Monitor the receipt of the file on the agent
- When it is received, delete the dummy rule at the event processor

Missing or empty event monitoring configuration file

You have received a MONMAN trace message on a workstation, similar to this:

```
MONMAN:INFO:=== DEPLOY ===> ERROR reading the zip file
               /home/f_edwa3/monconf/deployconf.zip.
               It is empty or does not exist".
```

Cause and solution:

The Tivoli Workload Scheduler agent on a workstation monitors for events using a configuration file. This file is created on the event processor, compressed, and sent to the agent. If a **switcheventprocessor** action is performed between the creation of the file on the old event processor and the receipt on the new event processor of the request for download from the agent, the file is not found on the new event processor, and this message is issued.

To resolve the problem, do the following:

- Create a dummy event rule that applies only to the affected workstation
- Perform a **planman deploy** to send the configuration file
- Monitor the receipt of the file on the agent
- When it is received, delete the dummy rule at the event processor

Events not processed in correct order

You have specified an event rule with two or more events that must arrive in the correct order, using the *sequence* event grouping attribute. However, although the events occurred in the required sequence the rule is not triggered, because the events arrived at the event processor in an order different from their creation order.

Cause and solution:

Events are processed in the order they arrive, not the order they are created. If they arrive in order different from the creation order, you will not get the expected result.

For example, consider a rule which is triggered if event A defined on workstation AA occurs before event B which is defined on workstation BB. If workstation AA loses its network connection before event A occurs, and does not regain it until after event B has arrived at the event processor, the event rule will not be satisfied, even though the events might have occurred in the correct order.

The solution to this problem is that if you need to define a rule involving more than one event, use the *set* event grouping attribute, unless you can be certain that the events will arrive at the event processor in the order they occur.

The stopeventprocessor or switcheventprocessor commands do not work

You have run **stopeventprocessor** or **switcheventprocessor** but the command has failed. The log indicates a communication problem.

Cause and solution:

If you issue the **stopeventprocessor** command from a workstation other than that where the event processor is configured, the command uses the command-line client, so the user credentials for the command-line client must be set correctly.

Similarly, if you use **switchevtprocessor**, it also uses the command-line client, so the user credentials for the command-line client must be set correctly also in this case.

Event rules not deployed with large numbers of rules

You have run **planman deploy** (or the equivalent action from the Tivoli Dynamic Workload Console), with a very large number of event rules, but the command has failed. The log indicates a memory error.

Cause and solution:

A large number of event rules requires a Java heap size for the application server larger then the default. In this context, a large number would be 10 000 or more. Doubling the default size should be sufficient.

Full details of how to do this are described in the *Tivoli Workload Scheduler: Administration Guide* in the section on *Increase application server heap size* in the *Performance* chapter.

Preventing problems with disk usage, process status, and mailbox usage

You can use event-driven workload automation (EDWA) to monitor the health of the Tivoli Workload Scheduler environment and to start a predefined set of actions when one or more specific events take place. You can prevent problems in the Tivoli Workload Scheduler environment by monitoring the filling percentage of the mailboxes, the status of Tivoli Workload Scheduler processes, and the disk usage of the Tivoli Workload Scheduler file system.

Full details of how to do this are described in the *Tivoli Workload Scheduler: Administration Guide*, as follows:

- section on *Monitoring the disk space used by Tivoli Workload Scheduler* in the *Data maintenance* chapter
- sections on *Monitoring the size of Tivoli Workload Scheduler message queues* and *Monitoring the status of Tivoli Workload Scheduler processes* in chapter *Network administration*

See also “Configuring traces for the Tivoli Workload Scheduler agent” on page 13.

Problems using the "legacy" global options

This section describes problems that might occur when running Tivoli Workload Scheduler with the "legacy" global options set. The "legacy" global options are those that have the word "Legacy" in their option name in **optman**. Use them if you want to maintain certain Tivoli Workload Scheduler behaviors as they were in previous versions of Tivoli Workload Scheduler.

- "Time zones not resolving correctly with enLegacyStartOfDayEvaluation set"
- "Dependencies not processed correctly when enLegacyId set"

Time zones not resolving correctly with enLegacyStartOfDayEvaluation set

You are using Tivoli Workload Scheduler with the `enLegacyStartOfDayEvaluation` and `enTimeZone` options set to *yes* to convert the `startOfDay` time set on the master domain manager to the local time zone set on each workstation across the network. You submit a job or job stream with the **at** keyword, but the job or job stream does not start when expected.

Cause and solution:

Add the **absolute** keyword to make sure that the submission times are resolved correctly. The **absolute** keyword specifies that the start date is based on the calendar day rather than on the production day.

Dependencies not processed correctly when enLegacyId set

You are using Tivoli Workload Scheduler in a network which includes agents running on versions older than 8.3, but managed by a version 8.3 or later master domain manager, with the `enLegacyId` option set to *yes*, to enable the use of the former job stream ID format. When you create multiple instances of a job stream as pending predecessors, errors caused by identification problems at submission time are given.

Cause and solution:

There is no workaround to this other than to upgrade the agents to the level of the master domain manager.

Managing concurrent accesses to the Symphony file

This section contains two sample scenarios describing how Tivoli Workload Scheduler manages possible concurrent accesses to the `Symphony` file when running **stageman**.

Scenario 1: Access to Symphony file locked by other Tivoli Workload Scheduler processes

If Tivoli Workload Scheduler processes are still active and accessing the `Symphony` file when **stageman** is run, the following message is displayed:

```
Unable to get exclusive access to Symphony.
Shutdown batchman and mailman.
```

To continue, stop Tivoli Workload Scheduler and rerun **stageman**. If **stageman** aborts for any reason, you must rerun both **planman** and **stageman**.

Scenario 2: Access to Symphony file locked by stageman

If you try to access the plan using the command-line interface while the Symphony is being switched, you get the following message:

```
Current Symphony file is old. Switching to new Symphony.
Schedule mm/dd/yyyy (nnnn) on cpu, Symphony switched.
```

Miscellaneous problems

The following problems might occur:

An error message indicates that a database table, or an object in a table, is locked

An error message indicates that a function cannot be performed because a table, or an object in a table, is locked. However, the table or object does not appear to be locked by another Tivoli Workload Scheduler process.

Cause and solution:

The probable cause is that a user has locked the table by using the database command-line or GUI:

DB2 Just opening the DB2 GUI is sufficient to lock the database tables, denying access to all Tivoli Workload Scheduler processes.

Oracle If the Oracle command-line is opened without the auto-commit option, or the GUI is opened, Oracle locks all tables, denying access to all Tivoli Workload Scheduler processes.

To unlock the table close the command-line or GUI, as appropriate.

Note: Tivoli Workload Scheduler provides all of the database views and reports you need to manage the product. You are strongly recommended to not use the facilities of the database to perform any operations, including viewing, on the database tables.

Command line programs (like composer) give the error "user is not authorized to access server"

You launch CLI programs (like composer) but when you try and run a command, the following error is given:

```
user is not authorized to access server
```

Cause and solution:

This problem occurs when the user running the command has a null password. Composer, and many of the other Tivoli Workload Scheduler CLI programs cannot run if the password is null.

Change the password of the user and retry the operation.

The rmstdlist command gives different results on different platforms

The rmstdlist command on a given UNIX platform gives results that differ from when it is used on other platforms with the same parameters and scenario.

Cause and solution:

This is because on UNIX platforms the command uses the *-mtime* option of the **find** command, which is interpreted differently on different UNIX platforms.

To help you determine how the *-mtime* option of the **find** command is interpreted on your workstation, consider that the following command:

```
<TWA_home>/TWS/bin/stdlist/rmstdlist -p 6
```

gives the same results as these commands:

```
find <TWA_home>/TWS/stdlist/ -type d ! -name logs ! -name traces -mtime +6 -print
find <TWA_home>/TWS/stdlist/logs/ -type f -mtime +6 -print
find <TWA_home>/TWS/stdlist/traces/ -type f -mtime +6 -print
```

Look at your operating system documentation and determine how the option works.

The rmstdlist command fails on AIX with an exit code of 126

The rmstdlist command on AIX fails with an exit code of 126 and no other error message.

Cause and solution:

This could be because there are too many log files in the `stdlist` directory.

On AIX, you should regularly remove standard list files every 10-20 days. See the usage instructions in the *Tivoli Workload Scheduler: User's Guide and Reference* for full details.

Question marks are found in the stdlist

You discover messages in the log or trace files that contain question marks. For example the following (the message has been split over several lines to make it more readable - the question marks are highlighted to make them more obvious):

```
10:20:02 03.02.2008|BATCHMAN:+ AWSBHT057W
Batchman has found a non-valid run number in the Symphony
file for the following record type: "Jt" and object:
"F235011S3_01#???[(),(0AAAAAAAAAAAAAZD)].A_7_13 (#J18214)".
```

Cause and solution:

This problem occurs when the process that needs to write the log message cannot obtain the job stream name. For example, when a job stream is dependent on a job stream that is not in the current plan (Symphony file). The process writes "???" in place of the missing job stream name.

The message contains the job stream ID (in the above example it is the string in the second set of parentheses: (0AAAAAAAAAAAAAZD). Use the job stream ID to identify the instance of the job stream, and take any action suggested by the message that contained the question marks.

A job with a "rerun" recovery job remains in the "running" state

You have run a job specifying a recovery job using the "rerun" recovery method. The original job fails, but when the recovery job starts the original job shows that the recovery action has been completed successfully, but remains in the "running" state.

Cause and solution:

This problem would occur if the recovery job was specified to run on a different workstation and domain from the original job. The original job is then unable to detect the state of the recovery job, so it cannot determine if the recovery job has finished or what state it finished in.

To resolve the problem for the specific job that is still in "running" state, you must manually stop the job.

To avoid the recurrence of the problem specify the "rerun" recovery action on the same workstation in the same domain.

Job statistics are not updated daily

Job statistics are not updated daily, as they were with versions prior to version 8.3.

Cause and solution:

Job statistics are updated by **JnextPlan**. If you are running **JnextPlan** less frequently than daily, the statistics are only updated when **JnextPlan** is run.

A job stream dependency is not added

A dependency is added to a job stream instance and the job stream is saved. When the list of dependencies is reopened, the new dependency is not present.

Cause and solution:

This occurs when a job stream instance already has the maximum number (40) of dependencies defined. Normally, an error message would alert you to the limit, but the message might not be displayed if there is a delay propagating the Symphony updates across the network or if your update coincided with updates by other users.

Incorrect time-related status displayed when time zone not enabled

You are using Tivoli Workload Scheduler in an environment where nodes are in different time zones, but the time zone feature is not enabled. The time-related status of a job (for example, "Late") is not reported correctly on workstations other than that where the job is being run.

Cause and solution:

Enable the time zone feature to resolve this problem. See *Tivoli Workload Scheduler: User's Guide and Reference* to learn more about the time zone feature. See *Tivoli Workload Scheduler: Administration Guide* for instructions on how to enable it in the global options.

Completed job or job stream not found

A job or job stream that uses an alias has completed but when you define a query or report to include it, the job or job stream is not included.

Cause and solution:

Jobs and job streams in *final* status are stored in the archive with their *original* names, not their aliases, so any search or reporting of completed jobs must ignore the aliases.

Variables not resolved after upgrade

After upgrading to version 8.5, global variables are not resolved.

Cause and solution:

During the upgrade to version 8.5, all the security file statements relating to your global variables were copied by the install wizard into a default variable table in the new security file. Global variables are disabled in version 8.5, and can only be used through the variable tables. If you subsequently rebuilt the security file using the output from your *previous* **dumpsec** as input to the new **makesec**, you will have overwritten the security statements relating to your default variable table, so no user has access to the default variable table.

If you have a backup of your security file from prior to when you ran **makesec**, run **dumpsec** from that, and merge your old **dumpsec** output file with your new one, as described in the upgrade procedure in the *Tivoli Workload Scheduler: Planning and Installation Guide*.

If you do not have a backup, create the default variable table security statement, following the instructions about configuring the security file in the *Tivoli Workload Scheduler: Administration Guide*.

Default variable table not accessible after upgrade

After upgrading to version 8.5, your default variable table is not accessible by any user.

Cause and solution:

This problem has exactly the same Cause and solution: as the preceding - see “Variables not resolved after upgrade” on page 106.

Local parameters not being resolved correctly

You have scheduled a job or job stream that uses local parameters, but the parameters are not resolved correctly.

Cause and solution:

One reason for this could be that one or both of the files where the parameters are stored have been deleted or renamed.

Check that the following files can be found in the *TWA_home*/TWS directory:

```
parameters
parameters.KEY
```

These files are required by Tivoli Workload Scheduler to resolve local parameters, so they must not be deleted or renamed. Fix the problem as follows:

1. If the files have been renamed, rename them to the original names.
2. If the files have been deleted, recreate them, using the **parms** utility.
3. To make the changes effective, restart the application server, using the **stopappserver** and **startappserver** commands.

Log files grow abnormally large in mixed environment with version 8.4 or higher master domain manager and 8.3 or lower agents

The problem occurs in mixed environments where Tivoli Workload Scheduler agents version 8.3 or earlier run under a master domain manager version 8.4 or later. The problem is that the older version agents do not correctly handle the Tivoli Workload Scheduler events generated by the features added by version 8.4 and later, such as Event Driven Workload Automation (monman), Workload Service Assurance (critical path), and WebSphere Application Server manager (appservman). This may cause random execution, duplication of Tivoli Workload Scheduler events or dumping of Tivoli Workload Scheduler event records type "00" that flood the log files.

The cure to this problem is to install on your older version agents the corresponding fix pack containing the fix for APAR IZ62730.

An alternative to installing the fix pack on your agents is to apply the following workaround on your version 8.4 or later master domain manager, provided your master runs one of the following product versions:

- 8.4 with fix pack 5 or later
- 8.5 with fix pack 1 or later
- 8.5.1 with fix pack 1 or later

Follow these steps:

1. Disable the Event Driven Workload Automation (EDWA) feature
   ```
   - optman chg ed=no
   ```
2. Check that EDWA is actually disabled
   ```
   - optman  ls
   >>>>>   enEventDrivenWorkloadAutomation / ed = NO
   ```
3. Shut down Tivoli Workload Scheduler and WebSphere Application Server
4. Delete the `Mailbox.msg` file because it contains messages related to stopping the `appservman` process
5. Enable new behavior of `appservman` by adding to the `localopts` file the following key:
   ```
   Appserver disable send event = yes
   ```
6. Start up Tivoli Workload Scheduler
7. Check that the broadcast of newer product versions (8.4 and later) events is actually disabled by looking for the following message in the `<TWS_home>/stdlist/traces/TWSMERGE.log`: *"Broadcasting of Appservman events is disabled"*

If you cannot find this message, the reason is that your master is not patched with the fix pack version listed above. If this is the case, you can run the following recovery procedure (but this will preclude `appservman` from starting):

1. Shut down Tivoli Workload Scheduler and WebSphere Application Server
2. Delete the `Mailbox.msg` file because it contains messages related to the start up of `appservman`
3. Start up WebSphere Application Server without the `appservman` process:
   ```
   <TWSHOME>/wastools/StartWas.sh -direct
   ```
4. Start up Tivoli Workload Scheduler without the *appservman* process
   ```
   Startup -noappsrv
   ```

The master domain manager is now ready to create a plan without the Event Driven Workload Automation. You can wait for the next `JnextPlan` or run:

```
JnextPlan -for 000
```

If you have a mix of version 8.3 and version 8.4 agents, follow these steps:

1. Unlink and shut down only the version 8.4 agents
2. Check that no Tivoli Workload Scheduler processes are running
   ```
   ps -fu <TWS_user>
   ```
3. Delete the `Mailbox.msg` file because it contains messages related to the `monman` process:
4. Disable the `monman` process from starting by modifying the following key in the `localopts` file:
   ```
   autostart monman = no
   ```
5. Restart Tivoli Workload Scheduler

Chapter 6. Troubleshooting dynamic workload broker

This chapter provides information that is useful in identifying and resolving problems with the dynamic workload broker component. It includes the following sections:

- "Diagnostic tools"
- "Maintaining an audit trail" on page 110
- "Tuning the rate of job processing" on page 118
- "Troubleshooting common problems" on page 120

Diagnostic tools

This section gives an overview of the diagnostic tools available for dynamic workload broker. The following topics are described:

- "Log and trace files"
- "Activating traces for Tivoli Dynamic Workload Broker"
- "Activating logs for Job Brokering Definition Console" on page 110

Log and trace files

The logs and traces produced by dynamic workload broker are in most part included in the log and trace files of Tivoli Workload Scheduler. In addition, the files listed in Table 6 are also created.

Table 6. Locations of log and trace files

Component	Path	Trace files	Log files	Content
Dynamic workload broker	*TWA_home* /ewas/Profiles/ twaprofile/logs/twaserver*N* *N* is the number of the TWA instance.	native_stderr.log native_stdout.log serverStatus.log startServer.log stopServer.log SystemErr.log trace.log	SystemOut.log	Additional dynamic workload broker log files
Tivoli Workload Scheduler agent	*TWA_home* /TWS/stdlist/JM	JobManager_trace.log ita_trace.log	JobManager_message.log ita_message.log	Log and trace files
	TWA_home /TWS/stdlist/JM/ JOBMANAGER-FFDC/*yy-mm-dd*/		JobManager_message.log	Processing error log file
Job Brokering Definition Console	*user's home directory*/ jd_workspace/.metadata/tivoli/ JBDC/logs	trace.log	msg.log, msg_cbe.log	Trace files
	$TEMP/TWA/jbdc851	trace_installation.log trace_installation_ xml.log	msg_installation.log	Installation log and trace files

Activating traces for Tivoli Dynamic Workload Broker

Use the `changeTraceProperties` command. It is explained in "Setting the traces on the embedded WebSphere Application Server for the major Tivoli Workload Scheduler processes" on page 15.

Activating logs for Job Brokering Definition Console

By default, logging is disabled. To generate log files, you must enable tracing in the **Preferences** dialog box.

To enable logging, perform the following steps:

1. Select **Preferences** in the Windows menu. The **Preferences** dialog box is displayed.
2. Optionally, specify a path and name for the log file in the **Log file directory** field.
3. Select the **Enable logging to console** check box.

The logs are saved in the directory indicated in Table 6 on page 109.

Maintaining an audit trail

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.

The auditing logs are created in XML format and can be viewed with a standard text editor or parsed using third-party utilities.

You can also view the auditing logs using the Log and Trace Analyzer (LTA), a component of the IBM Autonomic Computing Toolkit. In general, the Log and Trace Analyzer is used for importing and correlating different logs generated by different products. The Log and Trace Analyzer can be very useful in correlating dynamic workload broker auditing logs with other logs from different sources, such as databases (DB2, Oracle), WebSphere Application Server, and the operating system. See “Analyzing log files with Log Analyzer” on page 18 for details.

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

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 7. 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

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 quotes (') 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 desired 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.

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 7 on page 111.

Action
Represents the action that is being taken. Each auditable event supports a different set of possible actions. See Table 7 on page 111. The Action type contains the following element:

Table 8. Elements in Action type

Element name	Element description	Always returned in the output
Action	The action type that is being taken on the dynamic workload broker object.	Yes

ObjectInfoList

Represents a list of dynamic workload broker objects. The `ObjectInfoList` type contains the following element:

Table 9. 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 10. 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

Table 10. Elements in ObjectInfo type (continued)

Element name	Element description	Always returned in the output
objectSubmitterType	The type of the component which submitted the operation. The component is one of the following: • Tivoli 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 11. 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 12. 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 13. Elements in UserInfo type

Element name	Element description	Always returned in the output
UserInfo	The username provided to dynamic workload broker for authentication.	Yes

Performing 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>
```

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 Tivoli Workload Scheduler: Scheduling Workload Dynamically* or *IBM Tivoli 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>
```

Tuning the rate of job processing

The processing of jobs submitted for dynamic scheduling is handled by the two subcomponents of dynamic workload broker, job dispatcher and resource advisor, through a mechanism of queues and a cache memory. Job dispatcher uses a system of queues into which jobs are placed according to their processing status and thus transmitted to the resource advisor. Resource advisor uses a system of time slots during which it takes a number of jobs from the job dispatcher and allocates them to the resources that will run them.

The `JobDispatcherConfig.properties` and `ResourceAdvisorConfig.properties` configuration files are tuned to suit most environments. However, if your environment requires a high job throughput or if jobs are processed too slowly, you can add the parameters listed below to the specified configuration files and provide customized values. The configuration files are created for dynamic workload broker at installation time and are documented in *IBM Tivoli Workload Scheduler: Administration Guide*.

By default, the parameters listed below are not listed in the configuration files to prevent unwanted modifications. Only expert administrators should set these parameters.

JobDispatcherConfig.properties

MaxProcessingWorkers

Job dispatcher queues the submitted jobs according to their processing status. By default the following 3 queues are already specified:

```
Queue.actions.0 = cancel, cancelAllocation, completed, cancelOrphanAllocation
Queue.actions.1 = execute, reallocateAllocation
Queue.size.1 = 20
Queue.actions.2 = submitted, notification, updateFailed
```

Each queue is determined by the keywords:

Queue.actions.*queue_number*

Specifies the jobs added in this queue based on their processing status. The *queue_number* identifies the queue and ranges from 0 to 9. You can specify a maximum of 10 queues. The following table shows the entire list of process statuses you can specify in the queues.

Table 14. Job processing status to queue jobs for dispatching

activated	cancel	cancelAllocation
cancelJobCommand	cancelOrphanAllocation	childActivated
childCompleted	childDeactivated	childStarted
completed	deleteJobCommand	execute
getJobLogCommand	getJobPropertiesCommand	holdJobCommand
notification	reallocateAllocation	reconnect
resumeJobCommand	submitJobCommand	submitted
updateFailed	-	-

Unspecified job processing statuses are automatically placed in queue 0.

Queue.size.*queue_number*

Specifies the number of threads available to the queue identified by *queue_number*. You can specify 1 to 100 threads for each queue you define. The default is the number specified for `MaxProcessingWorkers`.

`MaxProcessingWorkers` specifies the default number of concurrent threads available to each queue. Each job dispatcher queue uses `MaxProcessingWorkers` threads, unless otherwise specified in `Queue.size.`*`queue_number`*. The `MaxProcessingWorkers` default is 10. Of the three default queues shown above, only queue 1 has its size specified to 20 threads (or workers). Queues 0 and 2 use the default defined in `MaxProcessingWorkers` (10 threads).

For example, in a test scenario with 250K jobs submitted through the dynamic workload broker workstation, the job allocation queues are re-configured as follows:

```
# Override default settings
Queue.actions.0 = cancel, cancelAllocation, cancelOrphanAllocation
Queue.size.0 = 10
Queue.actions.1 = reallocateAllocation
Queue.size.1 = 10
Queue.actions.2 = updateFailed
Queue.size.2 = 10

# Relevant to jobs submitted from
# dynamic workload broker workstation, when successful
Queue.actions.3 = completed
Queue.size.3 = 50
Queue.actions.4 = execute
Queue.size.4 = 50
Queue.actions.5 = submitted
Queue.size.5 = 50
Queue.actions.6 = notification
Queue.size.6 = 50

# Default for every queue size
MaxProcessingWorkers=10
```

Tune this parameter carefully to avoid impairing product performance.

HistoryDataChunk

Specifies the number of jobs to be processed at the same time when moving job data to the archive database. This is applicable only to a DB2 RDBMS. This parameter prevents an overload on the job dispatcher. The unit of measurement is jobs. The default value is 1000 jobs.

ResourceAdvisorConfig.properties

MaxAllocsPerTimeSlot

Specifies the number of requests for job allocation to be processed for each time slot. The default value is 100 requests per time slot. By default, each time slot lasts 15 seconds. Increasing this number causes the resource advisor to process a higher number of resource allocation requests per time slot with consequent processor time usage. This also allows the processing of a higher number of jobs per time slot. Decreasing this number causes the resource advisor to process a lower number of resource allocation requests per time slot resulting in a smoother processor usage and slower job

submission processing. You can also modify the time slot duration using the `TimeSlotLength` parameter available in this file.

MaxAllocsInCache
Specifies the number of requests for job allocation submitted by job manager to the resource advisor and stored in its cache. This number should be substantially higher than the value specified in the `MaxAllocsPerTimeSlot` parameter. The default value is 5000 allocation requests. Increasing this number causes the resource advisor to process a potentially higher number of resource reservations per time slot with consequent processor time usage. This also allows the processing of a higher number of jobs. Decreasing this number causes the resource advisor to process a lower number of resource reservations per time slot resulting in lower processor usage and slower job submission processing. For optimal performance, this value should be at least 10 times the value specified in the `MaxAllocsPerTimeSlot` parameter.

Troubleshooting common problems

The following problems could be encountered with dynamic workload broker:

- "Dynamic workload broker cannot run after the Tivoli Workload Scheduler database is stopped"
- "Getting an OutofMemory exception when submitting a job"

Dynamic workload broker cannot run after the Tivoli Workload Scheduler database is stopped

Dynamic workload broker cannot run as long as the database is down. When the database is up and running again, restart dynamic workload broker manually with the `startBrokerApplication` command. The command is described in *IBM Tivoli Workload Scheduler: Administration Guide*.

Getting an OutofMemory exception when submitting a job

If you get the following message after you submit a job for dynamic scheduling:

`The job with ID` *`job ID`* `failed to start. The error is "unable to create new native thread".`

you must tune a property of the scheduling agent.

The property is named **ExecutorsMinThreads** and is located in the *`TWA_home`*`/ITA/bin/JobManager.ini` file. Its default value is 38 but if this error occurs, you must decrease it to reduce the number of threads created when the job is launched.

The `JobManager.ini` file is described in the *IBM Tivoli Workload Scheduler: Administration Guide*.

Chapter 7. Troubleshooting Tivoli Dynamic Workload Console problems

This section describes the problems which could occur while using the Tivoli Dynamic Workload Console:

Note: For the troubleshooting run time scenarios impacting the Tivoli Dynamic Workload Broker environment, refer to the *Tivoli Dynamic Workload Broker: Installation and Configuration guide*

The problems are described in these groups:

Troubleshooting connection problems

The following problems could occur with the connection to the engine or the database:

The engine connection does not work

You define an engine connection, you verify that the values entered for the engine connection are correct, and then you click **Test Connection**. The test fails and a connection error message is returned.

Cause and solution:

Assuming that system_A is where you installed the Tivoli Dynamic Workload Console, and system_B is where you installed Tivoli Workload Scheduler, follow these verification steps to investigate and fix the problem:

1. Verify that there is no firewall between the two systems as follows:
   a. Make sure the two systems can ping each other. If you are trying to connect to a z/OS engine you must check that the system where the Tivoli Dynamic Workload Console is installed and the system where the Tivoli Workload Scheduler z/OS connector is installed can ping each other.
   b. Make sure you can telnet from system_A to system_B using the port number specified in the engine connection settings (for example, 31117 is the default port number for distributed engine).

   If either of these two steps fails then there might be a firewall preventing the two systems from communicating.
2. Check if you can connect using the **composer** command line interface, or the Job Scheduling Console (JSC) to the Tivoli Workload Scheduler engine on system_B using the same credentials specified in the engine connection. If you cannot, then check if the user definition on system_B and the user authorization specified in the Tivoli Workload Scheduler security file are correct.
3. If you are using LDAP or another User Registry on the Tivoli Dynamic Workload Console make sure that:
   a. The connection to the user registry works.
   b. The User Registry settings specified on the Integrated Solutions Console in the **Security** menu under **Secure administration, applications, and infrastructure** are correct.
   c. You restarted both the Tivoli Dynamic Workload Console's, and Tivoli Workload Scheduler's WebSphere Application Servers, after configuring the User Registry

   For more information about how to configure the Tivoli Dynamic Workload Console to use LDAP or about how to test the connection to a User Registry, refer to the chapter on configuring user security in the *Tivoli Workload Scheduler: Administration Guide*.
4. If you set up to use Single Sign-On between the Tivoli Dynamic Workload Console and the Tivoli Workload Scheduler engine, make sure you correctly shared the LTPA_keys as described in the chapter on configuring SSL in the *Tivoli Workload Scheduler: Administration Guide*.

   Note: Make sure that you correctly shared the LTPA_keys also if you get errors AWSUI0766E and AWSUI0833E. The problem occurs when the realm values are the same for more than one Websphere Application Server (Tivoli Dynamic Workload Console, Tivoli Workload Scheduler z/OS connector, or Tivoli Workload Scheduler engine). These steps are usually described only when you configure the Single Sign On, but they are required also when you have the same realm. You have the same realm when you configure all WebSphere Application Servers with the same LDAP user registry and when you install all Websphere Application Servers on the same machine.

If this checklist does not help you in identifying and fixing your problem then activate tracing on the Tivoli Dynamic Workload Console by running the steps listed in "Activating traces in Tivoli Dynamic Workload Console" on page 12 (adding also the Java packages com.ibm.ws.security.*=all:com.ibm.tws.*=all), and on the Tivoli Workload Scheduler engine by running the following steps:

1. Connect as WebSphere Application Server administrator to the system where the Tivoli Workload Scheduler engine is located.
2. Edit the file *TWA_home*/wastools/TracingProps.properties, add the statement:

   ```
   tws_with_sec=com.ibm.ws.security.*=all:com.ibm.tws.*=all
   ```

   and then save your changes.
3. Run the following script to start tracing:

   ```
   <TWA_home>/wastools/changeTraceProperties.sh -user
   <TWS_user> -password <TWS_user_password> -mode tws_with_sec
   ```

   where <*TWS_user*> and <*TWS_user_password*> are the credentials of the Tivoli Workload Scheduler administrator.

Connect to the Tivoli Dynamic Workload Console again, test the connection to the Tivoli Workload Scheduler engine, and then check the information stored in the following trace logs:

- On the Tivoli Dynamic Workload Console:

  ```
  <TWA_home>/eWAS/profiles/tdwc_profile/logs/<tdwc_server>/trace.log
  ```

  Note: If you installed the Tivoli Dynamic Workload Console on the embedded version of WebSphere Application Server, the <*tdwc_server*> is, by default, tdwcserver.
- On the Tivoli Workload Scheduler engine:

  ```
  <TWA_home>/eWAS/profiles/<TWS_profile>/logs/server1/trace.log
  ```

In these files you see the information about the error that occurred. If useful, compare the connection information stored in the traces with the information set for WebSphere Application Server security on both sides. Refer to the *Tivoli Workload Scheduler: Administration Guide* to list the information about the security properties.

Test connection takes several minutes before returning failure

You select an engine connection and click on **Test Connection** to check that the connection is working. The test takes several minutes to complete and then returns a failure.

Cause and solution:

When the **Test Connection** is run, the result is returned only after the timeout expires. The timeout for running the **Test Connection** operation cannot be customized. The connection failed because of one of the following reasons:

- The system where the Tivoli Workload Scheduler engine is installed is not active.
- The IP address or the hostname of the system where the Tivoli Workload Scheduler engine is installed was not correctly specified.
- A network firewall prevents the system where the Tivoli Dynamic Workload Console is installed and the system where the Tivoli Workload Scheduler engine is installed from communicating.

Check which of these reasons causes the communication failure, fix the problem, and then retry.

Engine connection settings are not checked for validity when establishing the connection

You incorrectly defined an engine connection to a distributed engine specifying a value for **Remote Server Name**. The **Remote Server Name** is not a valid setting for a connection to a distributed engine.

The check runs when you save the engine connection definition or when you run a test connection to that engine, but no exception about the incorrect setting is returned.

Cause and solution:

Whenever the test connection is run, only the mandatory fields for that specific type of engine, distributed rather than z/OS, are used to test the connection. Fields that are not mandatory, such as **Remote Server Name** for distributed engine connections are not taken into account.

Failure in testing a connection or running reports on an engine using an Oracle database

You test the connection to an engine by specifying the user credentials for an Oracle database, or you run a report on that engine connection. The operation fails and the following error message is displayed:

```
AWSUI0360E The JDBC URL is not configured on the selected engine,
so the reporting capabilities cannot be used.
          Contact the Tivoli Workload Scheduler administrator."
```

Cause and solution:

Make sure that the Tivoli Workload Scheduler administrator has updated the `TWSConfig.properties` file by adding the following key:

```
com.ibm.tws.webui.oracleJdbcURL
```

specifying the JDBC Oracle URL. For example:

```
com.ibm.tws.webui.oracleJdbcURL=jdbc:oracle:thin:@//9.132.235.7:1521/orcl
```

Rerun the operation after the `TWSConfig.properties` has been updated. For more information about showing and changing database security properties for Tivoli Workload Scheduler, refer to the *IBM Tivoli Workload Scheduler: Administration and Troubleshooting guide*.

Connection error when running historical reports or testing connection from an external instance of WebSphere Application Server

You try to test the connection to an engine where you Enable Reporting, or you try to run a historical report, the report fails and the following database connection error is saved to the WebSphere Application Server logs:

```
[date_and_time] 00000044 SystemErr R Exception in thread "WnTransactionThread-10"
java.lang.VerifyError:
class loading constraint violated (class: com/ibm/db2/jcc/c/p method:
getSQLJLogWriter()Lcom/ibm/db2/jcc/SQLJLogWriter;) at pc: 0
[date_and_time] 00000044 SystemErr R at java.lang.J9VMInternals.verifyImpl
                (Native Method)
[date_and_time] 00000044 SystemErr R at java.lang.J9VMInternals.verify
                (J9VMInternals.java:59)
```

```
[date_and_time] 00000044 SystemErr R at java.lang.J9VMInternals.verify
                (J9VMInternals.java:57)
[date_and_time] 00000044 SystemErr R at java.lang.J9VMInternals.initialize
                (J9VMInternals.java:120)
[date_and_time] 00000044 SystemErr R at com.ibm.db2.jcc.DB2Driver.connect
                (DB2Driver.java:163)
[date_and_time] 00000044 SystemErr R at java.sql.DriverManager.getConnection
                (DriverManager.java:562)
[date_and_time] 00000044 SystemErr R at java.sql.DriverManager.getConnection
                (DriverManager.java:186)
[date_and_time] 00000044 SystemErr R at
```

The Tivoli Dynamic Workload Console is installed on an external WebSphere Application Server together with other products using either DB2 or Oracle databases.

Cause and solution:

Because of a current WebSphere Application Server limitation, you must run these steps to run historical reports if your Tivoli Dynamic Workload Console is installed on an external WebSphere Application Server together with other products using either DB2 or Oracle databases.

1. Stop the WebSphere Application Server.
2. Access the directory:

   <TWA_home>/eWAS/systemApps/isclite.ear/TWSWebUI.war/WEB-INF/lib
3. Remove the following JDBC driver files:

   ```
   db2jcc.jar
   db2jcc_license_cu.jar
   ojdbc14.jar
   ```
4. Start WebSphere Application Server.

Note: This WebSphere Application Server limitation does not affect your activities if:

- You run Actual Production Details and Planned Production Details reports.
- You run operations that do not require to select Enable Reporting in the engine connection properties.

Connection problem with the engine when performing any operation

Whatever operation you try to run in the Tivoli Dynamic Workload Console, you get an error message saying that there is a connection problem with the engine.

Cause and solution:

Do the following steps:

1. Exit the Tivoli Dynamic Workload Console.
2. Restart the WebSphere Application Server.
3. Log in again to the Tivoli Dynamic Workload Console.

Continue with your activities on Tivoli Dynamic Workload Console.

Engine connection does not work when connecting to the z/OS connector (versions 8.3.x and 8.5.x)

If one of the following errors occurs when running the test connection, follow the steps described in the cause and solution section:

1. `AWSUI0766E Test connection to myengine: failed. AWSUI0833E The operation did not complete. There was a communication failure. The internal message is: AWSJZC093E The requested engine zserver is not defined.`
2. `AWSUI0766E Test connection to myengine : failed. AWSUI0833E The operation did not complete. There was a communication failure. The internal message is: A communication failure occurred while attempting to obtain an initial context with the provider URL: "corbaloc:iiop:ZOS_CONNECTOR_HOSTNAME:31127".`
3. `AWSUI0766E Test connection to myengine : failed. AWSUI0833E The operation did complete. There was a communication failure. The internal message is: EQQPH26E TME user ID missing in TME user to RACF userid mapping table: myuser@hostname1.test.com`

Cause and solution:

The possible causes for the case above are:

1. The name of the server startup job on host side must be defined on the z/OS connector before you perform the test connection from the TDWC.
2. The Websphere Bootstrap port is incorrect. Make sure that any bootstrap address information in the URL is correct and that the target name server is running. A bootstrap address with no port specification defaults to port 2809. Possible causes other than an incorrect bootstrap address or unavailable name server include the network environment and workstation network configuration.
3. The RACF user ID has not been defined in the mapping table on host side.

You can solve the problem as follows:

Environment description example

The environment is composed of a z/OS connector installed on the `hostname1.test.com`, a TDWC installed on either the same or another system, and a z/OS engine installed on the `hostname2.test.com(port 445)`.

Steps on the z/OS connector side

Define a connection from the z/OS connector to the host side by running the following script located in the directory `<ZCONN_INST_DIR>/wastools` and then restart WebSphere:

```
> createZosEngine -name zserver -hostName hostname2.test.com–portNumber 445
> stopWas
> startWas
```

where *zserver* is a logical name and can be changed to any other name.

Check the Bootstrap port by running the script showHostProperties.bat (sh) located in the directory `<ZCONN_INST_DIR>/wastools`.

Steps on the TDWC side

On the TDWC Web interface, define an engine connection from TDWC to the z/OS connector, as follows:

Engine name

Choose any name.

Engine Type
z/OS.

Host Name
Either `hostname1.test.com` or localhost.

Port Number
The z/OS connector Bootstrap port.

Remote Server Name
zserver (or the name you used in step 2 - createZosEngine).

User ID / Password
For example, the credentials you specified when installing TDWC and z/OS Connector (that is, the user that owns the TDWC and z/OS Connector instances). The user can be any user that is authenticated by the User Registry configured on the embedded WebSphere installed with the products.

Note: **Port Number** in version 8.5.x depends on which product is installed first. If TDWC is installed first, the Bootstrap port is 22809 and subsequent products installed on top of TDWC inherit that. If z/OS Connector is installed first, the Bootstrap port is 31217. If the z/OS connector version is 8.3 FPx, the default Bootstrap port is 31127.

Steps on the z/OS side

Make sure that user `myuser@hostname1.test.com` is defined in the RACF user ID mapping table on host side (USERMAP parameter in the SERVOPTS initialization statement).

Engine connection does not work when connecting to the z/OS connector V8.3.x or a distributed Tivoli Workload Scheduler engine V8.3.x

If one of the following errors occurs when running the test connection, follow the steps described in the cause and solution section:

1. `AWSUI0766E Test connection to myengine: failed. AWSUI0833E The operation did not complete.`

   ```
   Reason: AWSJCO005E WebSphere Application Server gives the following error:
   CORBA NO_PERMISSION 0x0 No; nested exception is:
   org.omg.CORBA.NO_PERMISSION: Trace from server: 1198777258 at host
   myhostname.com >>
   org.omg.CORBA.NO_PERMISSION: java.rmi.AccessException: ; nested exception is:
   com.ibm.websphere.csi.CSIAccessException:
   SECJ0053E: Authorization failed for /UNAUTHENTICATED while invoking (Bean)
   ejb/com/ibm/tws/zconn/engine/ZConnEngineHome
   getEngineInfo(com.ibm.tws.conn.util.Context):
   1 securityName: /UNAUTHENTICATED;accessID:
   UNAUTHENTICATED is not granted any of the required roles:
   TWSAdmin vmcid: 0x0 minor code: 0 completed: No . . .
   ```

2. `AWSUI0778E There was an authentication failure: the user name or password is incorrect.`

Cause and solution:

The symptoms above are caused because on the z/OS connector, or on the distributed engine side, the script **webui.sh** (bat) must be run to enable communication with the TDWC. Under the `wastools` directory of the home directory of the installation directory, run these commands:

```
./webui.sh -operation enable -user wasuser
-password waspwd -port soap_port
-pwdLTPA anypassword -server server1

./stopWas.sh -user wasuser -password waspwd

./startWas.sh
```

where:

user and password are those specified at installation time.

port is the WebSphere SOAP port (display it by running the command **showHostProperties.sh**).

pwdLTPA is any password used to export and encrypt the LTPA keys.

server is the WebSphere server name. The default is server1.

Engine connection does not work when connecting to distributed Tivoli Workload Scheduler engine V8.4 FP2

If one of the following errors occurs when running the test connection, follow the steps described in the cause and solution section:

```
AWSUI0766E Test connection to myengine: failed.
SECJ0053E: Authorization failed for /UNAUTHENTICATED while invoking
(Bean)ejb/com/ibm/tws/conn/engine/ConnEngineHome getEngineInfo
(com.ibm.tws.conn.util.Context):1 securityName:
/UNAUTHENTICATED;accessID: UNAUTHENTICATED is not granted any
of the required roles: TWSAdmin vmcid: 0x0 minor code: 0 completed: No
```

Cause and solution:

The problem is caused by a missing setting, which is already fixed in later versions of the engine. You can solve the problem by specifying on the engine instance the fully qualified hostname in the security.xml. Run the following steps to solve the problem:

1. Stop WebSphere on the engine using the command: <tws_install_dir>/wastools/stopWas.sh
2. Back up and then edit the following file (make sure that the editor does not change the formatting): <tws_install_dir>\eWAS\profiles\twaprofile\config\cells\DefaultNode\security.xml
3. Locate the line related to the CustomUserRegistry, for example:

   ```
   <userRegistries xmi:type="security:CustomUserRegistry"
   xmi:id="CustomUserRegistry_1203516338790"
   serverId="mywasadmin" serverPassword="{xor}Mj46LCstMA==" limit="0"
   ignoreCase="true" useRegistryServerId="true" realm=""
   customRegistryClassName="com.ibm.tws.pam.security.registry.
   PamUnixRegistryImpl"/>
   ```

4. Add the fully qualified hostname to the realm attribute, as in the following example:

   ```
   <userRegistries xmi:type="security:CustomUserRegistry"
   xmi:id="CustomUserRegistry_1203516338790"
   serverId="a840" serverPassword="{xor}Mj46LCstMA==" limit="0"
   ```

```
ignoreCase="true" useRegistryServerId="true"
realm="nc114040.romelab.it.ibm.com"
customRegistryClassName="com.ibm.tws.pam.security.registry.
PamUnixRegistryImpl"/>
```

5. Restart WebSphere on the engine using the command: `<tws_install_dir>/wastools/startWas.sh`

Note: If you have any problems when restarting WebSphere, restore the original `security.xml` and start again.

WebSphere does not start when using an LDAP configuration

The WebSphere startup fails and the `SystemOut.log` file contains one of the following messages with exceptions.

1.
```
SECJ0419I: The user registry is currently connected to the LDAP server
ldap://nc125088.romelab.it.ibm.com:389.
....
WSVR0009E: Error occurred during startup
com.ibm.ws.exception.RuntimeError: com.ibm.ws.exception.RuntimeError:
javax.naming.NameNotFoundException: [LDAP: error code 32 - No Such Object];
remaining name 'ou=asiapacific,dc=test,dc=it'
at com.ibm.ws.runtime.WsServerImpl.bootServerContainer(WsServerImpl.java:199)
at com.ibm.ws.runtime.WsServerImpl.start(WsServerImpl.java:140)
. . .
```
2.
```
SECJ0418I: Cannot connect to the LDAP server ldap://nc125088.romelab.it.
ibm.com:389.....
WSVR0009E: Error occurred during startup
com.ibm.ws.exception.RuntimeError: com.ibm.ws.exception.RuntimeError:
javax.naming.AuthenticationException: [LDAP: error code 49 - 80090308:
LdapErr: DSID-0C090334, comment: AcceptSecurityContext error, data 525,
vece...
```
3.
```
SECJ0270E: Failed to get actual credentials.
The exception is com.ibm.websphere.security.PasswordCheckFailedException:
No user AMusr1@test.it found
at com.ibm.ws.security.registry.ldap.LdapRegistryImpl.checkPassword
(LdapRegistryImpl.java:311)
at com.ibm.ws.security.registry.UserRegistryImpl.checkPassword
(UserRegistryImpl.java:308)
at com.ibm.ws.security.ltpa.LTPAServerObject.authenticate
(LTPAServerObject.java:766)
```
4.
```
SECJ0352E: Could not get the users matching the pattern AMusr1@test.it
because of the following exception javax.naming.CommunicationException:
nc1250881.romelab.it.ibm.com:389 [Root exception is
java.net.UnknownHostException:
nc1250881.romelab.it.ibm.com]
```

Cause and solution:

The answers to the problems are listed below. The answers refer to some of the security properties provided to the wastool script **changeSecurityProperties.sh** (bat).

1. Connect with an LDAP Browser to the LDAP server and verify that the `LDAPBaseDN` value is a valid Base Distinguished Name and ensure that the `LDAPServerId` value is an existing user for the `LDAPBaseDN`.
2. Ask the LDAP administrator for the user and password to perform LDAP queries and set them in the `LDAPBindDN` or `LDAPBindPassword` properties.
3. Connect with an LDAP Browser to the LDAP server and verify that the properties of a valid user match the properties specified in the `LDAPUserFilter`, and also ensure that these properties are congruent with the type of the value specified on the `LDAPServerId`. For example, the `objectCategory` must be an existing `objectClass` and if `LDAPServerId` is an e-mail address value, then the

property to use on the filter must be the "mail" coerently. A valid user filter for the example is: `(&(mail=%v)(objectCategory=user))`.

4. Ensure that the *LDAPHostName* is a valid existing host and that it can be reached on the network. A useful test is to try to telnet to that host on the *LDAPPort* specified.

After changing the properties as suggested in the above list, run the **changeSecurityProperties.sh** (bat) script again, providing a file containing the updated security properties. Then start WebSphere.

Troubleshooting performance problems

- "With a distributed engine the responsiveness decreases overtime"
- "Running production details reports might overload the distributed engine"
- "A "java.net.SocketTimeoutException" received"

With a distributed engine the responsiveness decreases overtime

When working with a distributed engine the responsiveness decreases overtime.

Cause and solution:

The problem might be related to multiple production plan report request running on that Tivoli Workload Scheduler engine, since those operations are CPU consuming. Ensure to wait until the report completion before running again other requests of the same kind.

Running production details reports might overload the distributed engine

The temporary directory on the distributed engine where the production details reports run, might be filled up.

Cause and solution:

The amount of memory used by the application server to extract the data varies depending on the number of objects to be extracted. For example, to extract 70 000 objects required almost 1 GB of RAM. By default the application server heap size is 512 MB, but it is possible to change this value as follows:

1. Log on to the Tivoli Workload Scheduler workstation as root.
2. Edit the following file:

   TWA_home`/eWAS/profiles/twaprofiles/config/cells/DefaultNode/nodes/DefaultNode/servers/twsserver<`*n*`>/server.xml`
3. Locate the option `maximumHeapSize` and set its value to at least 1024 (this value is expressed in Megabytes).
4. Stop and Start the application server.

As a general recommendation, use filters to avoid extracting huge production report files.

A "java.net.SocketTimeoutException" received

You are accessing the Tivoli Dynamic Workload Console with Internet Explorer 6.0, service pack 2, on a slow workstation (for example: Pentium 4, CPU 1.8 GHz) and are performing one of the following actions, which does not complete:

- You are querying objects in the plan, but on navigating through the result pages the browser hangs while drawing a result page, leaving the page with just the table header and footer shown and none of the result rows displayed. The hang of the browser can be resolved by clicking a button or link, but the missing data is not displayed.
- You are performing either a **Save**, **Edit**, or **Search** operation in the Workload Designer, which hangs for about 60 seconds and then displays one of these two error messages:

  ```
  AWSUI6171E  The operation could not be completed because the Tivoli
  Dynamic Workload Console server is unreachable. Possible causes are that
  the Tivoli Dynamic Workload Console server has been stopped or that your
  login authentication has expired or has become invalid.
  ```

  ```
  AWSUI6182E The operation could not be completed because an internal error
  occurred. The internal error is: the service name has not been provided.
  ```

Cause and solution:

What exactly causes the problem has not been ascertained (it might be a bug in Internet Explorer), but it can be resolved by increasing the value of one of the configurable timeouts in the application server.

Do the following:

1. Identify the instance of WebSphere Application Server running the Tivoli Dynamic Workload Console where this workstation normally connects to (if it connects to more than one, perform the procedure for all of them)
2. On that instance, edit the WebSphere Application Server configuration file "`server.xml`". The default location is

   ```
   <TWA_home>/eWAS/profiles/twaprofile/config/cells/DefaultNode/
        nodes/DefaultNode/servers/twaserver
   ```

3. Increase the value of the `persistentTimeout` of the `HTTPInboundChannel` related to the `WCInboundAdminSecure` chain section of the file. The default value is *30*, but for the given example (Pentium IV, CPU 1.8 GHz) a suggested value to set is *120*. An example using the relevant parts of a modified `server.xml` is as follows:
   a. Identify the `WCInboundAdminSecure` chain by looking in the chains section:

      ```
      <chains
       xmi:id="Chain_1226491023533"
       name="WCInboundAdminSecure"
       enable="true"
       transportChannels="TCPInboundChannel_1226491023530
                          SSLInboundChannel_1226491023530
                          HTTPInboundChannel_1226491023531
                          WebContainerInboundChannel_1226491023531"/>
      ```

      Note the value of the `HTTPInboundChannel`.

   b. Use the value of the `HTTPInboundChannel` to locate its entry:

      ```
      :<transportChannels
       xmi:type="channelservice.channels:HTTPInboundChannel"
       xmi:id="HTTPInboundChannel_1226491023531"
       name="HTTP_3"
       discriminationWeight="10"
       maximumPersistentRequests="100"
       keepAlive="true"
      ```

```
readTimeout="60"
writeTimeout="60"
persistentTimeout="120"
enableLogging="false"/>
```

Modify `persistentTimeout` as has already been done here.

4. Stop the instance of WebSphere Application Server using the **stopWas** command.
5. If a Tivoli Workload Scheduler component is also running under the same instance of the WebSphere Application Server, you need take no further action, as appservman will automatically restart the application server. Otherwise, use the **startWas** command.
6. Test the modified value to see if it has resolved the problem. If not, repeat the operation with a larger value, until the problem is resolved.

Troubleshooting user access problems

- "Wrong user logged in when using multiple accesses from the same system"
- "Unexpected user login request after having configured to use Single Sign-On"

Wrong user logged in when using multiple accesses from the same system

You try to access the Tivoli Dynamic Workload Console as *user2* using Firefox or Internet Explorer 7, where a connection as *user1* is already active in the same browser. In the case of Firefox the problem occurs if *user1* is active in *any* other Firefox window or tab. In Internet Explorer 7 the problem only occurs if the other user is active in a different tab of the same browser instance. But in both cases the result is the same: the browser logs you in to the Tivoli Dynamic Workload Console as *user1* instead of *user2*.

Cause and solution:

This is a browser limitation. If you have an active connection through Internet Explorer 7 to the Tivoli Dynamic Workload Console, and you want to open another session on the same system, you need only to open a different browser window. If the active connection is on Firefox, however, you must use a different browser. For a list of supported browsers, refer to the Tivoli Dynamic Workload Console System Requirements Document.

Unexpected user login request after having configured to use Single Sign-On

It might happen that, after running successfully all the steps required to configure the Single Sign-On between the Tivoli Dynamic Workload Console and a Tivoli Workload Scheduler engine, when you try to test the connection or run a task on that engine, you are unexpectedly prompted to enter your user credentials to connect. This behavior means that the Single Sign-On method is not working properly on that engine.

Cause and solution:

Make sure that the *application_server*/profiles/*profile_name*/config/cells/*cell_name*/security.xml files of both the Tivoli Dynamic Workload Console and the Tivoli Workload Scheduler engine have identical values assigned to the *realm*

field of the *security:LDAPUserRegistry* section. This setting belongs to the WebSphere Application Server profile configuration. For example, even though you ran all the required steps to configure the Single Sign-On, it might not work if you set `realm="myHost.myDomain:389"` on the Tivoli Dynamic Workload Console and `realm="myHost:389"` on the Tivoli Workload Scheduler engine.

Troubleshooting problems with reports

The output of a report run on Job Statistics View shows -1 in the Average CPU Time and Average Duration fields

You run a report accessing the Job Statistics Database View, such as Job Run Statistics or a Custom SQL report, and the output shows the value -1 in Average CPU Time and Average Duration fields.

Cause and solution:

The historical report, regardless of what kind of report you run (for Jobs, Workstations, or Custom SQL), reads in the database the information about the previous production plan run. If some fields in a database view are empty, the value returned in the report output is -1. This means that if you run **JNextPlan** for the first time, and then you run for example the Job Run Statistics report, the value of Average CPU Time and Average Duration fields is -1.

Run **JNextPlan** again, or wait for the *final* job stream to run, to populate the database views and get values different from -1.

The output of report tasks is not displayed in a browser with a toolbar installed

You tested that the connection to the database set in the engine connection works properly but, after you run a report task, no window opens in your browser to display the task results. You have a third-party toolbar installed on your browser.

Cause and solution:

A third-party toolbar (such as Yahoo! or Google or similar) installed on top of the browser might conflict with the correct operation of the Tivoli Dynamic Workload Console reporting feature. To make the reporting feature work correctly you must uninstall the toolbar and then rerun the report task.

WSWUI0331E error when running reports on an Oracle database

You try to run a report on an engine connection where an Oracle database has been referenced. The report task fails and the following error is displayed:

```
WSWUI0331E SQL validate failure.The database internal message is:ORA-00942:
           table or view does not exist
```

If you try to run an SQL query statement in the Oracle database on the same table or view using the userid specified for the database connection in the engine connection properties, the query runs successfully.

Cause and solution:

On Oracle databases only, you must run these steps, as Oracle database administrator, to allow the database user specified in the engine connection properties to run reports from the Tivoli Dynamic Workload Console:

1. Assign to the database user the "CREATE TABLE" Oracle System privilege.
2. Run the following script:

   On Windows
   TWA_home\TWS\dbtools\oracle\script\dbgrant.bat

   On UNIX:
   TWA_home/dbtools/oracle/script/dbgrant.sh

CSV report looks corrupted on Microsoft Excel not supporting UTF8

You run a report asking to save the result in a CSV file. When you open the CSV file using Microsoft Excel, the content of the file looks corrupted.

Cause and solution:

To bypass this problem, make sure that the version of Microsoft Excel you are using supports the UTF8 character set. If it does not, install a more recent version that supports UTF8. Then, follow these steps to correctly open CSV reports from Microsoft Excel:

1. Open Microsoft Excel.
2. In the **Data** menu entry, select **Import External Data** and then **Import Data**.
3. Select the CSV file saved and click **Open**.
4. In the field File Origin, select **UTF8**.

Insufficient space when running production details reports

When running production details reports the temporary directory on the Tivoli Workload Scheduler engine where the reports run, could be filled up.

Cause and solution:

You need to free some space in the temporary directory on the Tivoli Workload Scheduler engine before continuing to work on that engine.

Troubleshooting other problems

Data not updated after running actions against monitor tasks results

After running an action on a list of objects returned from running a monitor task the list is not updated.

Cause and solution:

The scheduling objects lists are not automatically updated after running actions. Click on the **Refresh** button to update the list of objects.

"Session has become invalid" message received

You try to use the Tivoli Dynamic Workload Console user interface, your working session closes, and you get the following warning:

```
Session has become invalid
Your session has become invalid. This is due to a session timeout, an administrator
has logged you out, or another user has invalidated your session by logging on with
the same User ID.
```

Cause and solution:

Check which reason among those listed in the warning has occurred, solve the issue, and then log in again to continue your working session.

If the session expired because either the HTTP session or the Lightweight Third Party Authentication (LTPA) session timeout was exceeded, you might decide to customize the timeout settings to values that are appropriate for your environment.

For instructions on how to do this, see the topic on session timeout settings in the *Performance* chapter of the *Tivoli Workload Scheduler: Administration Guide*.

Actions run against scheduling objects return empty tables

After running a monitor task, you run an action against the scheduling objects listed in the result table, but you get, as a result of the action, an empty table or window, and no error message is displayed. This occurs regardless of which action you try to run against the listed scheduling objects.

Cause and solution:

Check if the connection with the Tivoli Workload Scheduler engine where you run the task failed by doing the following:

- In the **Configuration** window select Scheduler Connections.
- Select in the list the engine used to run the browse task and click **Test Connection**.

Note: The user ID you use to connect to the Tivoli Dynamic Workload Console must belong either to the **TWSWEBUIAdministrator** or to the **TWSWEBUIConfigurator** groups to test the engine connection.

If the connection with the Tivoli Workload Scheduler engine is not active, ask the Tivoli Workload Scheduler administrator to restart the connection as described in the *IBM Tivoli Workload Scheduler Reference Guide*, and then rerun the action.

If the connection with the Tivoli Workload Scheduler engine is active, then, on that engine, check that:

- The Tivoli Workload Scheduler user running the command to list scheduling objects is authorized to do so. For more information about how to set user authorization, refer to the *IBM Tivoli Workload Scheduler Reference Guide*.
- The global property *enListSecChk* is set to enable on the Tivoli Workload Scheduler master domain manager. For more information about how to set global properties, refer to the *IBM Tivoli Workload Scheduler Planning and Installation Guide*.

Then rerun the action.

Default tasks are not converted into the language set in the browser

An existing user logs in to the Tivoli Dynamic Workload Console using a browser where the language set is different from the language that was set in the browser the first time he logged in. In the Manage Tasks window, the default tasks are not translated into the new language.

Cause and solution:

The default tasks are created, using the current language set in the browser, when the new user logs into the Tivoli Dynamic Workload Console for the first time. To have the default tasks translated into a different language, the WebSphere Application Server administrator must create a new Tivoli Dynamic Workload Console user, and use that to login to the Tivoli Dynamic Workload Console for the first time using a browser configured with the requested language. By doing this the default tasks are created using the requested language.

"Access Error" received when launching a task from the browser bookmark

A Tivoli Dynamic Workload Console task has been saved in the list of bookmarks of the browser. You try to launch the task using the bookmark but you receive the following error message:

"User does not have access to view this page, use the browser back button to return to previous page."

Cause and solution:

You have not the necessary role required to run the task. To run a task you must have a role that allows you to access the Tivoli Dynamic Workload Console panels that are relevant to the type of task you need.

For more information about setting roles to work with the Tivoli Dynamic Workload Console, see the Administration Guide, under the section about Configuring new users to access Tivoli Dynamic Workload Console

After Tivoli Workload Scheduler upgrade from version 8.3 to version 8.5 some fields in the output of reports show default values (-1, 0, unknown, regular)

After migrating Tivoli Workload Scheduler from version 8.3 to version 8.5, the output on the Tivoli Dynamic Workload Console of reports run on old migrated jobs show default values for the new fields introduced since version 8.3.

Cause and solution:

This is not a problem or a limitation but the result of migrating data from old tables to new tables containing newly created fields. After migration, it is necessary to assign a value to the new fields introduced since version 8.3 for job runs that occurred before migrating. The values assigned by default to these new fields are:

For job run statistic reports:

Table 15. Default settings for new job run statistic reports

Value	Field
0	Number of "Long Duration" job runs
0	Number of "Suppressed" job runs
0	Number of "Started Late" job runs
0	Number of "Ended late" job runs
0	Total Reruns
-1	Average CPU Time
-1	Average Duration

For job run history reports:

Table 16. Default settings for new job run history reports

Value	Field
unknown	Workstation Name (Job Stream)
-1	Started Late (delay hh:mm)

Table 16. Default settings for new job run history reports (continued)

Value	Field
-1	Ended Late (delay hh:mm)
-1	Estimated Duration (hh:mm)
No	Long Duration
Regular	Run Type
-1	Iteration Number
0	Return Code
0	Job Number
unknown	Login

The validate command run on a custom SQL query returns the error message AWSWUI0331E

You are creating a Custom SQL report, and you run the **Validate** command to check your query. The validate fails and the following error message is returned:

```
AWSWUI0331E The SQL query could not be validated. The database internal message is:
[ibm][db2][jcc][10103][10941] Method executeQuery cannot be used for update.
```

Cause and solution:

The validate failure is caused by a syntax error in the query statement, for example, a typing error, such as:

```
sele Workstation_name,Job_name,Job_start_time from MDL.JOB_HISTORY_V
 where Workstation_name like 'H%'
```

In this query, *sele* is written in place of *select*.

Verify the SQL query is correct and, optionally, try to run the same query from the DB2 command line to get additional details.

If you close the browser window, processing threads continue in the background

You perform an action or make a selection and immediately close the browser window. You expect that processing terminated but the messages stored in the `SystemOut.log` file show that processing continued in the background.

Cause and solution:

This is normal behavior for any WEB application, when the client browser is closed no notification is delivered to the server according to the HTTP protocol specifications. This is the reason why the last triggered thread continues to process even after the browser window was closed. You do not need to run any action, just allow the thread to end.

The list of Available Groups is empty in the Enter Task Information window

You are creating a task, and you notice that in the Enter Task Information the list of Available Groups is empty. You are using an LDAP user registry.

Cause and solution:

Log into the Integrated Solutions Console as administrator and check the advanced LDAP configuration settings are correct as follows:

1. In the Navigation tree click on **Security**.
2. Click on **Secure administration, applications, and infrastructure**.
3. Check that the **Available realm definitions** field is set to **Standalone LDAP registry**.
4. Click on **Configure**.
5. Click on **Advanced Lightweight Directory Access Protocol (LDAP) user registry settings** under **Additional Properties**.
6. Verify that the settings for groups and users are correct for your configuration.

For more information about how to set these values, refer to: http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/topic/com.ibm.websphere.express.doc/info/exp/ae/usec_advldap.html

JVM failure when working with the Tivoli Dynamic Workload Console on a Red Hat Enterprise Linux (RHEL) Version 5 system

When working with the Tivoli Dynamic Workload Console on a Red Hat Enterprise Linux Version 5 system, a user might see the error "Failed to find VM - aborting"

Cause and solution:

Red Hat Enterprise Linux Version 5 has a new security feature named 'Security Enhanced Linux', or SELinux for short. A weaker version of SELinux was included in Red Hat Enterprise Linux Version 4, and was disabled by default. Red Hat Enterprise Linux Version 5 defaults SELinux to enabled. SELinux helps to keep the host secure from certain types of malicious attacks. However, the default settings have been known in many cases to prevent Java from running properly.

To fix this issue, you can choose one of the following options:

- Configure SELinux so that it knows that the Tivoli Dynamic Workload Console Java related processes are acceptable to run.
- Change the mode of SELinux to *Permissive* by entering `setenforce 0` on the command line. SELinux will be fully enabled again the next time the system is rebooted or if `setenforce 1` is entered on the command line.

Missing daylight saving notation in the time zone specification on Tivoli Dynamic Workload Console 8.4 Fix Pack 1 and later

When using Tivoli Dynamic Workload Console 8.4, the time zone is displayed using the *Daylight Saving*, or *Summer* notation, for example:

```
Europe/Paris (Central European Summer Time, GMT+1:00)
```

Starting from Tivoli Dynamic Workload Console 8.4 Fix Pack 1, the *Summer* notation is no longer displayed and the time zone is expressed as follows:

```
Europe/Paris (Central European Time, GMT+1:00)
```

Cause and solution:

This is just a change in the standard time zone notation and does not affect the time conversion mechanisms. You can ignore this difference.

Unresponsive script warning with Firefox browser

When opening the Workload Designer with Firefox, the following warning message might appear:

```
Warning: Unresponsive script
A script on this page may be busy, or it may have stopped responding.
You can stop the script now, or you can continue to see if the script will complete.
```

Cause and solution:

This is caused by a Firefox timeout. If prompted with this warning message, choose the "Continue" option.

This behavior of Firefox is ruled by its `dom.max_script_run_time` preference, which determines the timeout that the browser must wait for before issuing the warning. The default value is 10 seconds, and might be changed to another value according to your needs.

To change this value, type `about:config` in the address field of the browser, scroll down to the preference, click it, change the value, and click **OK**.

Workload Designer does not show on foreground with Firefox browser

With Firefox, if you open the Workload Designer from a graphical view (with the `Open Job definition` or the `Open Job stream definition` commands), and the Workload Designer window is already open, this window might not be moved to the foreground.

Solution:

To fix this problem, change the Firefox settings as follows:

1. On the Firefox action bar select **Tools**, then **Options**, then **Content**, and finally **Advanced**
2. Enable the `Raise or lower windows` option

A "java.net.SocketTimeoutException" received

See the following scenario: “A "java.net.SocketTimeoutException" received” on page 130.

Language-specific characters are not correctly displayed in graphical views

When working with the graphical views some language specific characters might not be displayed correctly.

Cause and solution:

This might occur because the necessary language files have not been installed on the computer on which the Tivoli Dynamic Workload Console is running. To solve the problem, install the operating system language files on the system hosting the Tivoli Dynamic Workload Console.

Plan View panel seems to freeze with Internet Explorer version 7

When using Internet Explorer version 7, some actions performed in sequence might cause the **Plan View** browser window to freeze and stay frozen for about 5 minutes. After this timeframe the browser window resumes.

Cause and solution:

Action sequences that might cause this problem typically include opening multiple **Plan View** panels at the same time and refreshing the **Plan View** panels that were already open.

To avoid or limit this behavior add the Tivoli Dynamic Workload Console web site to the **Local intranet** security zone of Internet Explorer 7, with its default security level.

Plan View limit: maximum five users using the same engine

If you try to open the **Plan View** when five users are already concurrently using it, with the same engine, your request is rejected with the following error message: `AWSJCO136E No more than 5 users are allowed to perform this operation at the same time. The maximum number of concurrent requests has been reached: please try again later.`

Cause and solution:

The maximum number of users that can use the **Plan View** connected to the same engine is five.

If needed, you can modify this limit by editing the `com.ibm.tws.conn.plan.view.maxusers` property in the `TWSConfig.properties` file.

Chapter 8. Troubleshooting workload service assurance

This chapter provides information that is useful in identifying and resolving problems with the Workload Service Assurance feature. It includes the following sections:

Components involved in workload service assurance

Workload service assurance uses the following components to plan, monitor, and if necessary, intervene in the processing of jobs that are part of a critical network:

Planner

The planner component is triggered by the **JnextPlan** command. It includes a series of actions that result in the creation of the `Symphony` file on the master domain manager.

When workload service assurance is enabled, the planner calculates job streams and job networks, taking into consideration all "follows" dependencies in the new plan.

The planner then identifies all the jobs and job streams that are part of a critical network. These are jobs that are direct or indirect predecessors of a critical job. For each job, a critical start time is created and added to the `Symphony` file. It represents the latest time at which the job can start without putting the critical job deadline at risk.

The `Symphony` file is subsequently distributed to all agents.

Plan monitor

The plan monitor component is introduced with the workload service assurance feature. It runs in the WebSphere Application Server on the master domain manager and is responsible for keeping track of the job streams and job network and for updating it when changes to the plan occur either because of the normal running of jobs or because of manual operations.

The plan monitor holds the information that is required to monitor the progress of the jobs involved in a critical network, for example critical start, planned start, estimated start, and risk level. It changes these values in response to changes in the plan, identified by the **batchman** process running on the master domain manager and communicated to the plan monitor using the `server.msg` file.

The information maintained by the plan monitor can be viewed on the Tivoli Dynamic Workload Console in specialized views for critical jobs, allowing you easily to identify real and potential problems.

Agent processes (batchman and jobman)

Jobs in the critical network that are approaching the critical start time and have not started are promoted. The time at which the job is considered to be approaching its critical start time is determined by the global options setting `promotionOffset`.

The **batchman** process monitors the critical start time to determine if promotion is required and if so to schedule it at the highest job priority available in Tivoli Workload Scheduler. The **batchman** process also communicates with the **jobman** process, which is responsible for promoting the job at operating system level so that it receives more system resources when it starts. The operating system promotion is controlled by the local options settings jm promoted nice (UNIX) and jm promoted priority (Windows).

Exchange of information

Initially, the critical start time for jobs in the critical network is calculated by the planner and then recalculated, as required, by the plan monitor. Both of these components run on the master domain manager.

The critical start time is used by agents to determine when to promote a job. It is initially sent to the agent when the new `Symphony` file for the plan is distributed. Subsequent changes to critical start times are sent by the plan manager to agents using a Tivoli Workload Scheduler message. The agents update the local copy of the `Symphony` file.

The most common situations in which the plan monitor updates critical start times are:

- The Workload Designer functions on the Tivoli Dynamic Workload Console or the conman command are used to modify jobs in the critical network. For example, predecessor jobs are added or cancelled.
- When **JnextPlan** is run to create the plan extension that includes the critical job, jobs in the original plan might be predecessors of the critical job and so be part of the critical network. In this case, critical start times are calculated by the plan monitor and sent in messages to the agents. The local `Symphony` files are updated to include this information.

Common problems with workload service assurance

The following are problems that could occur when you are using Tivoli Workload Scheduler with workload service assurance enabled:

- "Critical start times not aligned"
- "Critical start times inconsistent" on page 145
- "Critical network timings changing unexpectedly" on page 145
- "A critical job is consistently late" on page 145

Critical start times not aligned

The values for critical start times in a critical network obtained from the appropriate conman commands on an agent are different from those displayed on the Tivoli Dynamic Workload Console.

Cause and solution:

Changes that affect the critical start times have been made to the plan since the Symphony file was sent to the agent. The changes are calculated on the master domain manager and sent to agents in messages. It is probable that the message has not reached the affected agent.

Check that the agent is active and linked to the master domain manager, either directly or by other domain managers.

Critical start times inconsistent

The values for critical start time in the chain of jobs in the critical network appears to be inconsistent. There are predecessor jobs that have critical start dates that are later than their successors.

Cause and solution:

This inconsistency occurs when critical start times are recalculated after some of the jobs in the critical network have completed. To optimize the calculation, new critical start times are only recalculated and updated for jobs that have not yet completed. The completed jobs retain the original critical start time. If a completed job is subsequently selected to be rerun, its critical start date will be recalculated.

Critical network timings changing unexpectedly

Timings for jobs in the critical network change even though there have been no user actions related to the timing of jobs.

Cause and solution:

Changes can be made to timings because of a plan extension or because of the submission of jobs or job streams.

A critical job is consistently late

A job that is defined as critical is consistently late despite promotion mechanisms being applied to it and its predecessors.

Cause and solution:

Using the successful predecessors task, compare the planned start, the actual start, and the critical start of all the predecessors of the late job. Check if any of them have time values that are too close together or have a planned start time that is later than the critical start time.

In such a case, you can:

- Consider changing the timings of these jobs. For example, postpone the deadline if possible, or if the deadline must be maintained anticipate the start of some of the jobs.
- Consider redesigning your job streams to optimize the paths that are causing delays.
- Increase the value of the `promotionOffset` global option, so that jobs are promoted earlier.
- On the workstations where jobs are tending to be late, increase the jm promoted nice (UNIX) and jm promoted priority (Windows) local options, so that promoted jobs receive more system resources.

A high risk critical job has an empty hot list

A job that is defined as critical is shown to be at high risk, but its hot list is empty.

Cause and solution:

This normally only occurs if you have designed a critical job or a critical predecessor with a conflict which means it will always be late, for example a start restriction after the critical job deadline. The hot list is empty if either the job or job stream that is causing the problem doesn't have its follows dependencies resolved, or the job stream that is causing the problem is empty.

The only solution is to examine the critical path in detail and determine where the problem lies. The steps to resolving this problem are the same as those documented in "A critical job is consistently late" on page 145.

Chapter 9. Troubleshooting the fault-tolerant switch manager

This section describes how to address the potential problems related to the use of the fault-tolerant switch manager.

It is divided into the following sections:

Event counter

The messages displayed in the log file concerning the event counter table are of three types:

- Messages that confirm the successful event counter initialization. No action is needed.
- Messages that the event counter reports related to problems not related to it. For example, they could reveal that the workstation received a message out of sequence. If action is required it does not impact the event counter.
- Messages that indicate that the event counter has failed. User action is needed to restore the counter.

This section concerns itself with this third type of messages.

Two processes can display this kind of error message:

Writer When an error message of this type is received from writer, the event counters stops. All messages received from the workstation which asked netman to activate writer, and from all its children, are ignored. This can lead to two situations:

- The workstation talking to writer is switched to a new manager. In this case the new manager asks for a counter table and receive a corrupt counter table. The replay protocol proceeds following the default behavior.
- Before the **switchmgr** operation can be performed, writer fails and is automatically restarted. In this case the counter mechanism partially repairs itself. New messages received by the process are stored in the counter, but the messages received by the writer from the moment the error message was displayed up to the point at which writer restarted are not tracked. The situation for a given workstation might be considered as reset only when the new instance of writer receives a message from it.

The situation is recovered after the next scheduled JnextPlan. If you need to recover more urgently, run **JnextPlan -for 0000** to refresh the Symphony file.

Mailman

When an error message of this type is received from mailman, the event counters stops. Mailman sets the IDs of all messages to 0. This means that there is a risk of duplication, because without the event counter, mailman is unable to properly sequence and process messages.

When the **switchmgr** is performed, and the new domain manager commences the replay protocol mechanism, for each message in the ftbox it looks at the position of the target workstation with respect to its own position in the tree:

- If the position of the target workstation in the workstation tree is higher than the new domain manager's (the workstation is either the domain manager or a full-status member of the parent domain of the domain where the **switchmgr** operation took place), the message is sent.
- If the position of the target workstation in the workstation tree is lower than the new domain manager's (the workstation either belongs to the domain where the **switchmgr** operation took place and it is not the new domain manager or is the domain manager or a full-status member of one of the child domains), the message is *not* sent.

The situation is recovered after JnextPlan.

Ftbox

If, on a full-status agent, you receive an error message concerning the ftbox, it means that the fault-tolerant backup domain manager feature is not working properly on that agent. Do not make this agent the new domain manager.

To restore the correct functionality of the feature on the instance, solve the problem as described in the error message, and restart the agent.

Troubleshooting link problems

When troubleshooting a link problem, the analysis is started from the master domain manager. The loss of the "F" flag at an agent indicates that some link had a problem. The absence of a secondary link can be located by matching the "W" flags found on the full-status fault-tolerant agent on the other side.

Consider the network shown in Figure 1 on page 149, where the workstation ACCT_FS, which is a full-status fault-tolerant agent, is not linked:

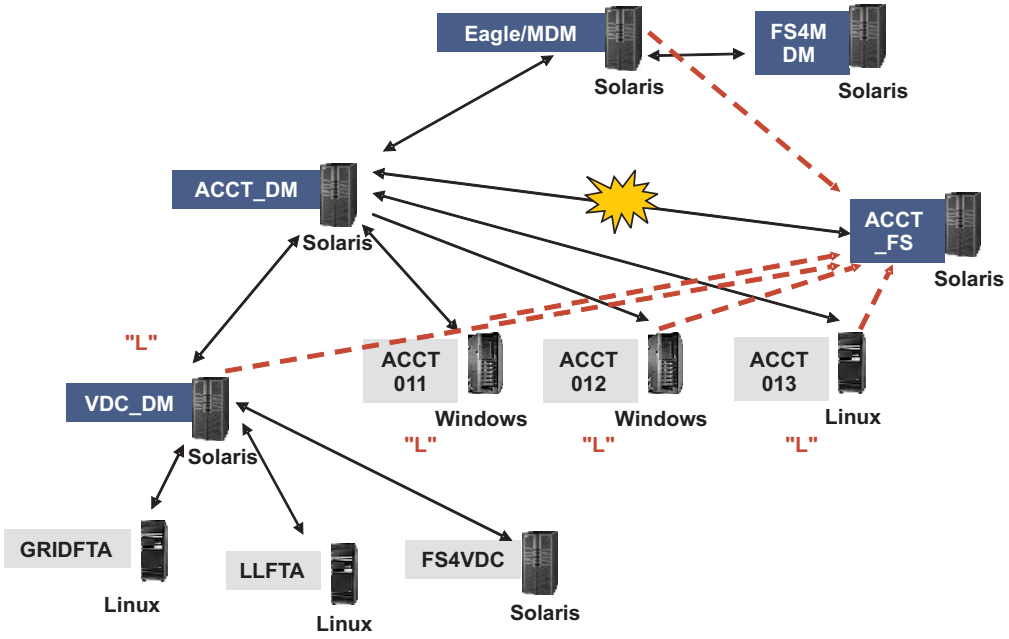


Figure 1. ACCT_FS has not linked

The key to Figure 1 is as follows (for those looking at this guide online or who have printed it on a color printer, the colors of the text and labels is indicated in parentheses, but if you are viewing it without the benefit of color, just ignore the color information):

White text on dark (blue) labels
CPUIDs of fault-tolerant agents in the master domain

Black text
Operating systems

Black text on grey labels
CPUIDs of standard agents in the master domain, or any agents in lower domains

Text (red) in "quotes"
Status of workstations obtained by running **conman sc @!@** at the master domain manager. Only statuses of workstations that return a status value are shown.

Black double-headed arrows
Primary links in master domain

Explosion
Broken primary link to ACCT_FS

Dotted lines (red)
Secondary links to ACCT_FS from the other workstations in the ACCT domain that could not be effected.

You might become aware of a network problem in a number of ways, but if you believe that a workstation is not linked, follow this procedure to troubleshoot the fault:

1. Use the command **conman sc @!@** on the master domain manager, and you can see that there is a problem with ACCT_FS, as shown in the example command output in Figure 2:

```
$ conman sc @!@
Installed for user 'eagle'.
Locale LANG set to "C"
Schedule (Exp) 01/25/05 (#365) on EAGLE.  Batchman LIVES.  Limit: 20, Fence: 0,
Audit Level: 1
sc @!@
CPUID      RUN   NODE     LIMIT FENCE    DATE   TIME  STATE  METHOD   DOMAIN
EAGLE      365  *UNIX MASTER   20    0   01/25/05 05:59   I J            MASTERDM
FS4MDM     365   UNIX FTA      10    0   01/25/05 06:57 FTI JW           MASTERDM
ACCT_DM    365   UNIX MANAGER  10    0   01/25/05 05:42 LTI JW           DM4ACCT
ACCT011    365   WNT  FTA      10    0   01/25/05 06:49 L I J            DM4ACCT
ACCT012    365   WNT  FTA      10    0   01/25/05 06:50 L I J            DM4ACCT
ACCT013    365   UNIX FTA      10    0   01/25/05 05:32 L I J            DM4ACCT
ACCT_FS    363   UNIX FTA      10    0                                   DM4ACCT
VDC_DM     365   UNIX MANAGER  10    0   01/25/05 06:40 L I J            DM4VDC
FS4VDC     365   UNIX FTA      10    0   01/25/05 06:55 F I J            DM4VDC
GRIDFTA    365   OTHR FTA      10    0   01/25/05 06:49 F I J            DM4VDC
GRIDXA     365   OTHR X-AGENT  10    0   01/25/05 06:49 L I J   gridage+ DM4VDC
LLFTA      365   OTHR FTA      10    0   01/25/05 07:49 F I J            DM4VDC
LLXA       365   OTHR X-AGENT  10    0   01/25/05 07:49 L I J   llagent  DM4VDC
$
```

Figure 2. Example output for **conman sc @!@** *run on the master domain manager*

2. From the ACCT_DM workstation run **conman sc**. In this case you see that all the writer processes are running, except for ACCT_FS. These are the primary links, shown by the solid lines in Figure 1 on page 149. The output of the command in this example is as shown in Figure 3:

```
$ conman sc
TWS for UNIX (SOLARIS)/CONMAN 8.5 (1.36.2.21)
Licensed Materials  Property of IBM
5698-WKB
(C) Copyright IBM Corp 1998,2001
US Government User Restricted Rights
Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM
Corp.
Installed for user 'dm010'.
Locale LANG set to "C"
Schedule (Exp) 01/25/05 (#365) on ACCT_DM.  Batchman LIVES.  Limit: 10, Fence: 0
, Audit Level: 1
sc
CPUID      RUN   NODE     LIMIT FENCE    DATE   TIME  STATE  METHOD   DOMAIN
EAGLE      365   UNIX MASTER   20    0   01/25/05 05:59 LTI JW           MASTERDM
ACCT_DM    365  *UNIX MANAGER  10    0   01/25/05 05:42   I J            DM4ACCT
ACCT011    365   WNT  FTA      10    0   01/25/05 06:49 LTI JW           DM4ACCT
ACCT012    365   WNT  FTA      10    0   01/25/05 06:50 LTI JW           DM4ACCT
ACCT013    365   UNIX FTA      10    0   01/25/05 05:32 LTI JW           DM4ACCT
ACCT_FS    363   UNIX FTA      10    0                                   DM4ACCT
VDC_DM     365   UNIX MANAGER  10    0   01/25/05 06:40 LTI JW           DM4VDC
$
```

Figure 3. Example output for **conman sc** *run on the domain manager*

3. From the ACCT_FS workstation run **conman sc**. In this case you see that there are no writer processes running. These are the secondary links, shown with the dashed lines in Figure 1 on page 149. The output of the command in this example is as shown in Figure 4:

```
$ conman sc
Installed for user 'dm82'.
Locale LANG set to "C"
Schedule (Exp) 01/24/05 (#364) on ACCT_FS.  Batchman LIVES.  Limit: 10, Fence: 0
, Audit Level: 1
sc @!@
CPUID      RUN   NODE     LIMIT FENCE    DATE   TIME  STATE  METHOD   DOMAIN
EAGLE      363   UNIX MASTER   20    0                      W           MASTERDM
FS4MDM     363   UNIX FTA      10    0                                  MASTERDM
ACCT_DM    363   UNIX MANAGER  10    0                                  DM4ACCT
ACCT011    363   WNT  FTA      10    0                                  DM4ACCT
ACCT012    363   WNT  FTA      10    0                                  DM4ACCT
ACCT013    363   UNIX FTA      10    0                                  DM4ACCT
ACCT_FS    363  *UNIX FTA      10    0                                  DM4ACCT
VDC_DM     363   UNIX MANAGER  10    0                                  DM4VDC
FS4VDC     363   UNIX FTA      10    0                                  DM4VDC
GRIDFTA    363   OTHR FTA      10    0                                  DM4VDC
GRIDXA     363   OTHR X-AGENT  10    0                          gridage+ DM4VDC
$
```

Figure 4. Example output for **conman sc** *run on the unlinked workstation*

4. If a network problem is preventing ACCT_FS from linking, resolve the problem.
5. Wait for ACCT_FS to link.
6. From the ACCT_FS workstation, run **conman sc @!@**. If the workstation has started to link, you can see that a writer process is running on many of the workstations indicated in Figure 1 on page 149. Their secondary links have now been made to ACCT_FS. The workstations that have linked have an "F" instead of their previous setting. This view also shows that the master domain manager has started a writer process running on ACCT_FS. The output of the command in this example is as shown in Figure 5 on page 152:

```
$ conman sc @!@
Installed for user 'dm82'.
Locale LANG set to "C"
Schedule (Exp) 01/24/05 (#364) on ACCT_FS.  Batchman LIVES.  Limit: 10, Fence: 0
, Audit Level: 1
sc @!@
CPUID      RUN   NODE     LIMIT FENCE    DATE   TIME  STATE  METHOD   DOMAIN
EAGLE      371   UNIX MASTER   20    0   01/25/05 10:16 F I JW           MASTERDM
FS4MDM     370   UNIX FTA      10    0                                   MASTERDM
ACCT_DM    371   UNIX MANAGER  10    0   01/25/05 10:03 LTI JW           DM4ACCT
ACCT011    369   WNT  FTA      10    0                                   DM4ACCT
ACCT012    371   WNT  FTA      10    0   01/25/05 11:03 F I JW           DM4ACCT
ACCT013    371   UNIX FTA      10    0   01/25/05 09:54 F I JW           DM4ACCT
ACCT_FS    371  *UNIX FTA      10    0   01/25/05 11:08 F I J            DM4ACCT
VDC_DM     371   UNIX MANAGER  10    0   01/25/05 10:52 F I JW           DM4VDC
FS4VDC     371   UNIX FTA      10    0   01/25/05 11:07 F I J            DM4VDC
GRIDFTA    371   OTHR FTA      10    0   01/25/05 11:01 F I J            DM4VDC
GRIDXA     371   OTHR X-AGENT  10    0   01/25/05 11:01 L I J   gridage+ DM4VDC
LLFTA      371   OTHR FTA      10    0   01/25/05 12:02 F I J            DM4VDC
LLXA       371   OTHR X-AGENT  10    0   01/25/05 12:02 L I J   llagent  DM4VDC
$
```

Figure 5. Example output for **conman sc @!@** *run on the unlinked workstation*

7. Another way of checking which writer processes are running on ACCT_FS is to run the command: **ps -ef | grep writer** (use Task Manager on Windows). The output of the ps command in this example is as shown in Figure 6:

```
$ ps -ef | grep writer
dm82  1363  616  0  06:43:11 ? 0:01  /usr/local/Tivoli/dm82/bin/write -- 2001  EAGLE MAILMAN UNIX 8.3 9
dm82  1317  616  0  06:42:21 ? 0:01  /usr/local/Tivoli/dm82/bin/write -- 2001  ACCT_DM MAILMAN UNIX 8.3 9
dm82  1337  616  0  06:42:25 ? 0:01  /usr/local/Tivoli/dm82/bin/write -- 2001  ACCT013 MAILMAN UNIX 8.3 9
dm82  1338  616  0  06:42:27 ? 0:01  /usr/local/Tivoli/dm82/bin/write -- 2001  VDC_DM MAILMAN UNIX 8.3 9
dm82  1364  616  0  06:51:48 ? 0:01  /usr/local/Tivoli/dm82/bin/write -- 2001  ACCT012 MAILMAN WNT 8.3 9
dm82  1336  616  0  06:42:24 ? 0:00  /usr/local/Tivoli/dm82/bin/write -- 2001  ACCT011 MAILMAN WNT 8.3 9
$
```

Figure 6. Example output for **ps -ef | grep writer** *run on the unlinked workstation*

8. To determine if a workstation is fully linked, use the **Status of all Workstations** list in the Job Scheduling Console or the equivalent option in the Tivoli Dynamic Workload Console. The option is supported in the Job Scheduling Console from feature level 1.3, fix pack #4 onwards. If you do not see the "FULLY LINKED" flag illustrated, you might not have updated the Tivoli Workload Scheduler Connector.

Common problems with the backup domain manager

The following problems could be encountered with the fault-tolerant backup domain manager (note that a backup domain manager is an agent with the *full status* attribute set):

- "The Symphony file on the backup domain manager is corrupted." on page 153
- "Processes seem not to have been killed on previous UNIX domain manager after running switchmgr" on page 153
- "In a scenario involving more than one switchmgr command, agent cannot relink" on page 153

The Symphony file on the backup domain manager is corrupted.

When switching to the backup domain manager from the master domain manager, the Symphony file on the backup domain manager might become corrupted.

Cause and solution:

The "thiscpu" variable in the `localopts` file does not match the workstation name. Change the variable to match the workstation name and the problem no longer occurs.

Processes seem not to have been killed on previous UNIX domain manager after running switchmgr

You want to use the switch manager facility. You first stop all Tivoli Workload Scheduler processes on the domain manager and then you run **switchmgr**, which completes successfully. However, after running **%sc @!@**, the J flag state is given for the domain manager where you stopped the processes.

Cause and solution:

When a **shutdown** command is sent to a workstation, some unexpected output might be shown by the status of the processes shown by conman, as follows:

- The J flag relative to the shut workstation remains active (no message indicating that jobman is not running can be transmitted because mailman is also not running).
- Conman output on the shutdown workstation is not up-to-date (the Symphony file is not updated on the shutdown workstation)
- The shutdown workstation seems linked from its father and son workstations (no unlink operation is run by the writers on the workstation that is shutting down)
- Both F or L flags might be displayed, depending on the messages processed by mailman before unlinking and stopping.

The correct link situation is restored as soon as a new link attempt is made to the workstation, either manually, or automatically (after 10 minutes).

The **shutdown** command must be sent only in critical situations (where a workstation is shutting down, for example).

To avoid these problems, precede the **shutdown** command with an **unlink @!@** or **stop** command.

In a scenario involving more than one switchmgr command, agent cannot relink

You have been using the **switchmgr** command to switch to backup master domain manager, and then back to the master domain manager, but an agent might not have relinked to the original master domain manager.

Cause and solution:

The complex interaction of variables, environments, network conditions, and linking and relinking events can sometimes prevent an agent from relinking correctly.

No events or messages are lost, you can repeat the use of **switchmgr**, if necessary, and the performance of the network is not normally impacted because one agent is out of communication.

If only one agent is involved the easiest solution is to manually relink it.

However, to avoid having to identify and specifically relink the non-linked agent or agents, you can, in any case, issue the following command, which automatically relinks all agents without needing to specifically identify the unlinked ones:

```
JnextPlan -for 0000
```

Chapter 10. Corrupt Symphony file recovery

This section describes how to diagnose and fix a corruption of the Symphony file.

Symphony file corruption is a rare event, and a potential corruption must be verified before taking action. Common symptoms are the following:

- A specific message informing you that the Symphony file is corrupt
- A shutdown of various processes (especially batchman) with error messages referring to problems with the Symphony file in the stdlist

The normal reason for the corruption of the Symphony file is a full file system. This can be avoided by regular monitoring of the file system where Tivoli Workload Scheduler is installed.

The procedure is different, depending on the location of the corrupt Symphony file:

- "Recovery procedure on a master domain manager"
- "Recovery procedure on a fault-tolerant agent or lower domain manager" on page 158

Recovery procedure on a master domain manager

If the `Symphony` file is corrupt on a master domain manager, it can be regenerated using the backup master domain manager.

The regeneration of the `Symphony` file causes some minor loss of data. The following procedure indicates what is lost.

The prerequisite for the procedure is that you have a backup master domain manager already available. A backup master domain manager is a fault-tolerant agent in the master domain with its `fullstatus` attribute set to *yes*.

Note: If you have not already created a backup master domain manager, the `Symphony` file cannot be recovered and the processing it contains is lost.

The procedure requires you to take the following steps on either the master domain manager or the backup master domain manager:

Note: The steps must be followed in strict order; each step description below is prefaced by the identification of the workstation on which it must be performed.

1. On the backup master domain manager, do the following:
   a. Issue the **switchmgr** command.
   b. Verify that the backup master domain manager is acting as the master domain manager.
2. From the new master domain manager set the job "limit" on the old master domain manager to "0", using **conman**, the Tivoli Dynamic Workload Console, or the Job Scheduling Console.

   This prevents jobs from launching.

3. On the original master domain manager do the following:
   a. Shut down all Tivoli Workload Scheduler processes
   b. Rename the `Sinfonia` file and the corrupt `Symphony` file (any names will do).
4. On the current master domain manager (previous backup master domain manager) do the following:
   a. Verify that it is linked to all agents *except* the old master domain manager.
   b. Shut down all Tivoli Workload Scheduler processes (unlink from all agents).
   c. Rename `Sinfonia` as `Sinfonia.orig`
   d. Copy `Symphony` to `Sinfonia`

      You now have identical `Symphony` and `Sinfonia` files.
5. On the original master domain manager do the following:
   a. Issue a **StartUp** from the operating system's command line, to start the **netman** process.
   b. Verify that the process remains active.
6. On the current master domain manager (previous backup master domain manager) do the following:
   a. Issue a **StartUp** from the operating system's command line, to start the **netman** process.
   b. Issue a **conman start**, or use the Tivoli Dynamic Workload Console or the Job Scheduling Console to start the current master domain manager.
   c. Issue a link to the original master domain manager.

      This action sends the Symphony file to the original master domain manager.
7. On the original master domain manager do the following:
   a. Verify that the Symphony file is present and is the correct size (same as on the current master domain manager (previous backup master domain manager)
   b. Verify that all Tivoli Workload Scheduler processes are active.
8. On the current master domain manager (previous backup master domain manager) verify that the original master domain manager is linked.
9. On the original master domain manager do the following:
   a. Set the job "limit" on the old master domain manager to the previous level, using **conman**, the Tivoli Dynamic Workload Console, or the Job Scheduling Console.

      Jobs can commence launching.
   b. Verify that the original master domain manager has the current job status for all agents.
   c. Issue the **switchmgr** command to switch control back to the original master domain manager.

Following this procedure some information is lost, in particular, any events that were suspended on the master domain manager when you started the recovery procedure.

If this procedure cannot be performed, try using the procedure described below, in "Alternative procedures for recovering the Symphony file on the master domain manager" on page 157.

Alternative procedures for recovering the Symphony file on the master domain manager

The following procedures can also be used to recover a corrupt Symphony file. They do not recover as much data as "Recovery procedure on a master domain manager" on page 155, but they might be useful if that procedure cannot be performed.

The procedure that makes use of `ResetPlan` might result in a more complete recovery, but it is more demanding in time since it scratches both the production and the preproduction plans. The preproduction plan will be created again based on the modeling information stored in the database when you later generate a new production plan. This means that the new production plan will contain all job stream instances scheduled to run in the time frame covered by the plan regardless of whether or not they were already in COMPLETE state when the plan was scratched.

You should first run the recovery procedure that makes use of `logman`. If you do not obtain satisfactory results, run the other one.

Neither procedure requires the use of a backup master domain manager.

Recovering with the use of the logman command

Follow these steps on the master domain manager :

1. Set the job "limit" to "0", using **conman**, the Tivoli Dynamic Workload Console, or the Job Scheduling Console.

   This prevents jobs from launching.
2. Shut down all Tivoli Workload Scheduler processes on the master domain manager.
3. Run **logman -prod** to update the preproduction plan with the information on the job streams in COMPLETE state.
4. Run **planman showinfo** and check for the first incomplete job stream instance.
5. Run **JnextPlan**, setting the **-from** parameter to the start time of the first incomplete job stream instance in the preproduction plan (acquired from the output of `planman showinfo`) and the **-to** parameter to the end date of your plan (or to the following day). Only incomplete job stream instances will be included in the new Symphony file.
6. Check the created plan and ensure that you want to run all the instances it contains, deleting those that you do not want to run.
7. Reset the job "limit" to the previous value. The Symphony file is distributed and production starts again.

Recovering with the use of the ResetPlan command

Follow these steps on the master domain manager :

1. Set the job "limit" to "0", using **conman**, the Tivoli Dynamic Workload Console, or the Job Scheduling Console.

   This prevents jobs from launching.
2. Shut down all Tivoli Workload Scheduler processes on the master domain manager.

3. Run **ResetPlan -scratch**.
4. Run **JnextPlan**, setting the **-from** and **-to** parameters to cover the period for which there are still outstanding jobs.
5. Check the created plan and ensure that you want to run all the instances it contains, deleting those that you do not want to run.
6. Reset the job "limit" to the previous value. The Symphony file is distributed and production recommences.

Recovery procedure on a fault-tolerant agent or lower domain manager

If the Symphony file is corrupt on a lower level domain manager, or on a fault-tolerant agent, it can be replaced.

Complete removal and replacement of the Symphony file causes some loss of data. The following procedure minimizes that loss and indicates what is lost.

The procedure involves two agents, the agent where the Symphony file is corrupt and its domain manager.

Note: Where the agent is a top level domain manager (below the master), or a fault-tolerant agent in the master domain, the manager is the master domain manager.

The procedure is as follows:

1. On the domain manager, unlink the agent which is having the Symphony file problem.
2. On the agent do the following:
   a. Stop the agent if it has not yet failed. You do not need to shut it down.
   b. Delete the Symphony and the Sinfonia files from the agent workstation. Alternatively you can move them to a different location on the agent workstation, or rename them.
3. On the domain manager do the following:
   a. Back up the Sinfonia file if you want to be able to restore the original situation after completion. This is not an obligatory step, and no problems have been reported from not performing it.
   b. Ensure that no agent is linking with the domain manager, optionally stopping the domain manager agent.
   c. Copy the domain manager's Symphony file to the Sinfonia file, replacing the existing version.
   d. Restart the domain manager agent if necessary.
   e. Link the agent and wait for the Symphony file to copy from the domain manager to the agent. The agent automatically starts.
   f. Optionally restore the Sinfonia file from the backup you took in step 3a. This restores the original situation, but with the agent now having an uncorrupted Symphony file. This is not an obligatory step, and no problems have been reported from not performing it.

Following this procedure some information is lost, in particular, the contents of the Mailbox.msg message and the tomaster.msg message queues. If state information about a job was contained in those queues, such that the Symphony file on the domain manager was not updated by the time the Sinfonia file is replaced (step 3c), that job is rerun. To avoid that event, add these steps to the procedure

immediately before step 3a on page 158:

1. Make a list of jobs that ran recently on the agent.
2. At the domain manager, change their states to either SUCC or ABEND, or even cancel them on the domain manager.

   Note: if you set the states of jobs to SUCC, or cancel them, any successor jobs would be triggered to start. Ensure that this is the acceptable before performing this action.

This way these jobs are not rerun.

Appendix A. Support information

If you have a problem with your IBM software, you want to resolve it quickly. This section describes the following options for obtaining support for IBM software products:

Using IBM Support Assistant

The IBM Support Assistant is a free, stand-alone application that you can install on any workstation. You can then enhance the application by installing product-specific add-on modules for the IBM products you use.

The IBM Support Assistant saves you time searching product, support, and educational resources. The IBM Support Assistant helps you gather support information when you need to open a problem management record (PMR), which you can then use to track the problem.

The product-specific add-on modules provide you with the following resources:
- Support links
- Education links
- Ability to submit problem management reports

The IBM Support Assistant Web site is at http://www.ibm.com/software/support/isa/. Use this site for the following:

- Obtain general information about the IBM Support Assistant
- Choose the IBM Support Assistant more appropriate for your needs, and perform the following actions:

  Download the IBM Support Assistant Lite for your product
  To quickly collect diagnostic files to solve problems faster. This is a special offering of the IBM Support Assistant that contains only the data collection component customized for a specific product. IBM Support Assistant Lite provides quick deployment of IBM Support Assistant's data collection tool. It is customized to automate product-specific data collection. You can run ISA Lite to do data collection for your product without ever installing ISA or your ISA add-on.

  Download and install the IBM Support Assistant Workbench V4.1 and the add-on for Tivoli Workload Scheduler V8.5.1
  To benefit from concurrent search, media viewer, guided troubleshooter, diagnostic tools, data collectors, service request submission, and other features. IBM Support Assistant can be customized for over 350 products.

  Note: To locate and download the add-on for a product, use the IBM Support Assistant's interface. Full instructions about how to use the application and add-on are provided within the interface. The

add-on for Tivoli Workload Scheduler V8.5.1 is indicated as add-on for V8.5 because the add-on is at release level. Add-ons are available at http://www.ibm.com/support/docview.wss?&uid=swg27013117).

If you cannot find the solution to your problem in the IBM Support Assistant, see "Searching knowledge bases."

Searching knowledge bases

You can search the available knowledge bases to determine if your problem was already encountered and is already documented.

Searching the local information center

IBM provides extensive documentation that you can install on your local computer or on an intranet server. You can use the search function of this information center to query conceptual information, instructions for completing tasks, and reference information.

The information center can be found online at http://publib.boulder.ibm.com/infocenter/tivihelp/v3r1/index.jsp?toc=/com.ibm.tivoli.itws.doc_8.5.1/toc.xml, from where you can download and install the information center locally, or on your intranet server.

Searching the Internet

If you cannot find an answer to your question in the information center, search the Internet for the latest, most complete information that might help you resolve your problem.

To search multiple Internet resources for your product, use the **Web search** topic in your information center. In the navigation frame, click **Troubleshooting and support ▸ Searching knowledge bases** and select **Web search**. From this topic, you can search a variety of resources, including the following:

- IBM technical notes (Technotes)
- IBM downloads
- IBM Redbooks®
- IBM developerWorks®
- Forums and newsgroups
- Google

Search the IBM support Web site

The IBM software support Web site has many publications available online, one or more of which might provide the information you require:

1. Go to the IBM Software Support Web site (http://www.ibm.com/software/support).
2. Select **Tivoli** under the **Select a brand and/or product** heading.
3. Select **IBM Tivoli Workload Scheduler** under **Select a product**, and click the "Go" icon:  . The Tivoli Workload Scheduler support page is displayed.

4. In the **IBM Tivoli Workload Scheduler support** pane click **Documentation**, and the documentation page is displayed.
5. Either search for information you require, or choose from the list of different types of product support publications in the **Additional Documentation support links** pane:
   - Information center
   - Manuals
   - IBM Redbooks
   - White papers

   If you click on **Information center** the Tivoli Workload Scheduler Information Center page opens, otherwise a search for the selected documentation type is performed, and the results displayed.
6. Use the on-screen navigation to look through the displayed list for the document you require, or use the options in the **Search within results for** section to narrow the search criteria. You can add **Additional search terms** or select a specific **Document type**. You can also change the sort order of the results (**Sort results by**). Then click the search icon to start the search:  .

To access some of the publications you need to register (indicated by a key icon beside the publication title). To register, select the publication you want to look at, and when asked to sign in follow the links to register yourself. There is also a FAQ available on the advantages of registering.

Obtaining fixes

A product fix might be available to resolve your problem. To determine what fixes are available for your IBM software product, follow these steps:

1. Go to the IBM Software Support Web site (http://www.ibm.com/software/support).
2. Select **Tivoli** under the **Select a brand and/or product** heading.
3. Select **IBM Tivoli Workload Scheduler** under **Select a product** and click the "Go" icon:  . The Tivoli Workload Scheduler support page is displayed.
4. In the **IBM Tivoli Workload Scheduler support** pane click **Download**, and the download page is displayed.
5. Either choose one of the displayed most-popular downloads, or click **View all download items**. A search for the downloads is performed, and the results displayed.
6. Use the on-screen navigation to look through the displayed list for the download you require, or use the options in the **Search within results for** section to narrow the search criteria. You can add **Additional search terms**, or select a specific **Download type**, **Platform/Operating system**, and **Versions**.

   Then click the search icon to start the search:  .
7. Click the name of a fix to read the description of the fix and to optionally download the fix.

For more information about the types of fixes that are available, see the *IBM Software Support Handbook* at http://www14.software.ibm.com/webapp/set2/sas/f/handbook/home.html.

Receiving support updates

To receive e-mail notifications about fixes and other software support news, follow these steps:

1. Go to the IBM Software Support Web site at http://www.ibm.com/software/support.
2. Click **My notifications** under the **Stay informed** heading in the upper-right corner of the page.
3. If you have already registered for **My support**, sign in and skip to the next step. If you have not registered, click **register now**. Complete the registration form using your e-mail address as your IBM ID and click **Submit**.
4. Follow the instructions on the page for subscribing to the information you require, at the frequency you require, for the products you require.

If you experience problems with the **My notifications** feature, you can obtain help in one of the following ways:

Online
Send an e-mail message to erchelp@ca.ibm.com, describing your problem.

By phone
Call 1-800-IBM-4You (1-888 426 4409).

Contacting IBM Software Support

IBM Software Support provides assistance with product defects.

Before contacting IBM Software Support, your company must have an active IBM software maintenance contract, and you must be authorized to submit problems to IBM. The type of software maintenance contract that you need depends on the type of product you have:

- For IBM distributed software products (including, but not limited to, Tivoli, Lotus®, and Rational® products, as well as DB2 and WebSphere products that run on Windows, or UNIX operating systems), enroll in Passport Advantage® in one of the following ways:

  Online

  Go to the Passport Advantage Web site at http://www.lotus.com/services/passport.nsf/ WebDocs/Passport_Advantage_Home and click **How to Enroll**.

  By phone

  For the phone number to call in your country, go to the IBM Software Support Web site support handbook contacts page at http://www14.software.ibm.com/webapp/set2/sas/f/handbook/contacts.html, and click **IBM Directory of worldwide contacts** or select your geographical area for a list of contacts.
- For customers with Subscription and Support (S & S) contracts, go to the Software Service Request Web site at https://www.software.ibm.com/webapp/set2/ssr.
- For customers with IBMLink, CATIA, Linux, S/390®, System i®, System p®, System z®, and other support agreements, go to the IBM Support Line Web site at http://www.ibm.com/services/us/index.wss/so/its/a1000030/dt006.
- For IBM eServer™ software products (including, but not limited to, DB2 and WebSphere products that run in System i, System p, and System z environments), you can purchase a software maintenance agreement by working directly with an IBM sales representative or an IBM Business Partner. For more information about support for eServer software products, go to the IBM Technical Support Advantage Web site at http://www.ibm.com/servers/eserver/techsupport.html.

If you are not sure what type of software maintenance contract you need, call 1-800-IBMSERV (1-800-426-7378) in the United States. From other countries, go to the contacts page of the *IBM Software Support Handbook* on the Web at http://www14.software.ibm.com/webapp/set2/sas/f/handbook/contacts.html and click the name of your geographic region for phone numbers of people who provide support for your location.

To contact IBM Software support, follow these steps:

1. "Determining the business impact" on page 166
2. "Describing problems and gathering information" on page 166
3. "Submitting problems" on page 166

Determining the business impact

When you report a problem to IBM, you are asked to supply a severity level. Therefore, you need to understand and assess the business impact of the problem that you are reporting. Use the following criteria:

Severity 1
The problem has a *critical* business impact. You are unable to use the program, resulting in a critical impact on operations. This condition requires an immediate solution.

Severity 2
The problem has a *significant* business impact. The program is usable, but it is severely limited.

Severity 3
The problem has *some* business impact. The program is usable, but less significant features (not critical to operations) are unavailable.

Severity 4
The problem has *minimal* business impact. The problem causes little impact on operations, or a reasonable circumvention to the problem was implemented.

Describing problems and gathering information

When describing a problem to IBM, be as specific as possible. Include all relevant background information so that IBM Software Support specialists can help you solve the problem efficiently. To save time, know the answers to these questions:

- What software versions were you running when the problem occurred?
- Do you have logs, traces, and messages that are related to the problem symptoms? IBM Software Support is likely to ask for this information.
- Can you re-create the problem? If so, what steps were performed to re-create the problem?
- Did you make any changes to the system? For example, did you make changes to the hardware, operating system, networking software, and so on.
- Are you currently using a workaround for the problem? If so, be prepared to explain the workaround when you report the problem.

Submitting problems

You can submit your problem to IBM Software Support in one of two ways:

Online
Click **Submit and track problems** on the IBM Software Support site at http://www.ibm.com/software/support/probsub.html. Type your information into the appropriate problem submission form.

By phone
For the phone number to call in your country, go to the IBM Software Support Web site support handbook contacts page at http://www14.software.ibm.com/webapp/set2/sas/f/handbook/contacts.html, and click **IBM Directory of worldwide contacts** or select your geographical area for a list of contacts.

If the problem you submit is for a software defect or for missing or inaccurate documentation, IBM Software Support creates an Authorized Program Analysis Report (APAR). The APAR describes the problem in detail. Whenever possible, IBM Software Support provides a workaround that you can implement until the

APAR is resolved and a fix is delivered. IBM publishes resolved APARs on the Software Support Web site daily, so that other users who experience the same problem can benefit from the same resolution.

Appendix B. Date and time format reference - strftime

Tivoli Workload Scheduler uses the *strftime* standard method for defining the presentation of the date and time in log files generated by CCLog. There is a parameter in the properties file of CCLog, where you define the format (see "Generating engine log files with CCLog" on page 8).

This parameter uses one or more of the following variables, each of which is introduced by a "%" sign, separated, if required, by spaces or other character separators.

For example, to define a date and time stamp that would produce the following (12-hour time, followed by the date) "7:30:49 a.m. - November 7, 2008", you would use the following definition:

```
%l:%M:%S %P - %B %e, %G
```

The full details of the parameters you can use are as follows:

Table 17. strftime date and time format parameters

Parameter	Description	Example
%a	The abbreviated weekday name according to the current locale.	Wed
%A	The full weekday name according to the current locale.	Wednesday
%b	The abbreviated month name according to the current locale.	Jan
%B	The full month name according to the current locale.	January
%c	The preferred date and time representation for the current locale.	
%C	The century number (year/100) as a 2-digit integer.	19
%d	The day of the month as a decimal number (range 01 to 31).	07
%D	Equivalent to %m/%d/%y. (This is the USA date format. In many countries %d/%m/%y is the standard date format. Thus, in an international context, both of these formats are ambiguous and must be avoided.)	12/25/04
%e	Like %d, the day of the month as a decimal number, but a leading zero is replaced by a space.	7
%G	The ISO 8601 year with century as a decimal number. The 4-digit year corresponding to the ISO week number (see %V). This has the same format and value as %y, except that if the ISO week number belongs to the previous or next year, that year is used instead.	2008
%g	Like %G, but without century, i.e., with a 2-digit year (00-99).	04
%h	Equivalent to %b.	Jan
%H	The hour as a decimal number using a 24-hour clock (range 00 to 23).	22
%I	The hour as a decimal number using a 12-hour clock (range 01 to 12).	07
%j	The day of the year as a decimal number (range 001 to 366).	008
%k	The hour (24-hour clock) as a decimal number (range 0 to 23); single digits are preceded by a blank. (See also %H.)	7

Table 17. strftime date and time format parameters (continued)

Parameter	Description	Example
%l	The hour (12-hour clock) as a decimal number (range 1 to 12); single digits are preceded by a blank. (See also %I.)	7
%m	The month as a decimal number (range 01 to 12).	04
%M	The minute as a decimal number (range 00 to 59).	58
%n	A newline character.	
%p	Either `AM' or `PM' according to the given time value, or the corresponding strings for the current locale. Noon is treated as `pm' and midnight as `am'.	AM
%P	Like %p but in lowercase: `am' or `pm' or a corresponding string for the current locale.	am
%r	The time in a.m. or p.m. notation. In the POSIX locale this is equivalent to `%I:%M:%S %p'.	07:58:40 am
%R	The time in 24-hour notation (%H:%M). For a version including the seconds, see %T below.	07:58
%s	The number of seconds since the Epoch, i.e., since 1970-01-01 00:00:00 UTC.	1099928130
%S	The second as a decimal number (range 00 to 61). the upper level of the range 61 rather than 59 to allow for the occasional leap second and even more occasional double leap second.	07
%t	A tab character.	
%T	The time in 24-hour notation (%H:%M:%S).	17:58:40
%u	The day of the week as a decimal, range 1 to 7, Monday being 1. See also %w.	3
%U	The week number of the current year as a decimal number, range 00 to 53, starting with the first Sunday as the first day of week 01. See also %V and %W.	26
%V	The ISO 8601:1988 week number of the current year as a decimal number, range 01 to 53, where week 1 is the first week that has at least 4 days in the current year, and with Monday as the first day of the week. See also %U and %W.	26
%w	The day of the week as a decimal, range 0 to 6, Sunday being 0. See also %u.	5
%W	The week number of the current year as a decimal number, range 00 to 53, starting with the first Monday as the first day of week 01.	34
%x	The preferred date representation for the current locale without the time.	
%X	The preferred time representation for the current locale without the date.	
%y	The year as a decimal number without a century (range 00 to 99).	04
%Y	The year as a decimal number including the century.	2008
%z	The time-zone as hour offset from GMT. Required to emit RFC822-conformant dates (using "%a, %d %b %Y %H:%M:%S %z").	-2
%Z	The time zone or name or abbreviation.	GMT
%%	A literal `%' character.	%

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this publication 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 publication. The furnishing of this publication does not give you any license to these patents. You can send license inquiries, in writing, to:

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

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

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement might not apply to you.

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

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

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

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
2Z4A/101
11400 Burnet Road
Austin, TX 78758 U.S.A.

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 publication 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.

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

Trademarks

IBM, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at http://www.ibm.com/legal/copytrade.shtml.

Intel is a trademark of Intel Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

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

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

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

Other company, product, and service names may be trademarks or service marks of others.

Index

Special characters

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

Program Number: 5698-WSH

Printed in USA

SC32-1275-10

Spine information:

IBM Tivoli Workload Scheduler **Version 8.5.1 (Revised May 2010)** **Troubleshooting Guide**