High Availability Cluster Environments

Version 9 Release 1
IBM Tivoli Workload Scheduler

High Availability Cluster Environments

Version 9 Release 1
Chapter 1. Types of high availability  1
Agent availability versus job availability  1
HACMP for AIX scenario - Backup domain manager  1

Chapter 2. Tivoli System Automation for Multiplatforms high availability  3

Chapter 3. Tivoli Workload Scheduler with Windows cluster  5
Windows cluster overview  5
Tivoli Workload Scheduler with Microsoft  5
Windows cluster environments  5
Prerequisite knowledge  5
Design limitations  5
Supported operating systems  6
Compatibility, upgrade, and coexistence  6
Security and Authentication  7
Tivoli Workload Scheduler Windows Cluster Enabler Components  7
Installation and configuration  9
Upgrading agents to a new version and available fix packs  15
twsClusterAdm command with examples of usage  18
Operating Tivoli Workload Scheduler in Windows cluster environment  29
Tivoli Workload Scheduler Cluster Administrator extension  33
Uninstalling Tivoli Workload Scheduler  34
Troubleshooting  35

Chapter 4. IBM Tivoli Workload Scheduler with HACMP  39
High-Availability Cluster Multi-Processing  39
Benefits  39
Physical components of an HACMP cluster  40
UNIX cluster overview  42
Prerequisite knowledge  43
Standby and takeover configurations  43
Design limitations  46
Supported configurations  46
Upgrading from previous versions of the product  50

Appendix. Resolving desktop heap size problems on workstations with more than three agents  53
Problem description  53
Solutions  54
Modify the shared heap buffer sizes  54
Configure the Tivoli Workload Scheduler Windows service to start as a local system account  54
Customize the desktop name so that it is reused  55
Implementing the solutions  55
Modify configuration of Windows service  55
Modify the Windows registry entries that determine the heap size  56
Modify localopts to supply a shared desktop name  56

Notices  57
Trademarks  58

Index  61
Figures

1. Main components of the Tivoli Workload Scheduler Cluster Enabler ........ 8
2. Installing in a cluster .......... 10
3. Clusters in a Tivoli Workload Scheduler network .......... 10
5. File Server example on Windows Server 2008 .......... 15
8. Cluster group example on Windows Server 2003 .......... 32
9. Localopts – Notepad .......... 33
10. New Properties Parameters tab .......... 34
11. Shared disk with mirror .......... 41
12. Active-Passive configuration in normal operation .......... 44
13. Failover on Active-Passive configuration .......... 44
14. Logical file system volumes .......... 45
15. Failover scenario .......... 45

© Copyright IBM Corp. 1999, 2013
Tables
About this publication

This publication describes how Windows and HACMP for AIX clusters fit into the topology of IBM® Tivoli Workload Scheduler. This publication also describes the enhancements to Tivoli Workload Scheduler to support the clustering and high-availability environment based on Microsoft Windows.

What is new in this release

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


What is new in this release for cluster support

The following additions have been made to the Tivoli Workload Scheduler support for the clustering and high-availability environment based on Microsoft Windows Server 2003 and Windows Server 2008:

- Two new generic options to enable monitoring of dynamic scheduling agent and Job Manager.
- A call to a new script to stop and start Tivoli Workload Scheduler.

Who should read this publication

This publication is intended for the following audience:

- Those who operate Tivoli Workload Scheduler in a UNIX and Linux cluster environment.
- IT Administrators or Tivoli® Workload Scheduler IT administrators – those who plan the layout of the Tivoli Workload Scheduler network.
- Installers – those who install the various software packages on the computers that make up the Tivoli Workload Scheduler network.

Publications

Full details of Tivoli Workload Scheduler publications can be found in Tivoli Workload Automation: Publications. This document also contains information about 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 Dynamic Workload Console, see the Accessibility Appendix in the IBM Tivoli Workload Scheduler User’s Guide and Reference.

Tivoli technical training

For Tivoli technical training information, refer to the following IBM Tivoli Education website:

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:

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

For more information about these three ways of resolving problems, see the appendix on support information in Tivoli Workload Scheduler: Troubleshooting Guide.
Chapter 1. Types of high availability

Server clusters are designed to keep resources (such as applications, disks, and file shares) available. Availability is a measure of the ability of clients to connect with and use a resource. If a resource is not available, clients cannot use it.

It is possible to contrast high-availability with fault-tolerance, as different benchmarks for measuring availability:

**Fault-tolerance**

Fault-tolerance is defined as 100% availability all of the time. Fault-tolerant systems are designed to guarantee resource availability.

**High-availability**

A high-availability system maximizes resource availability. A highly available resource is available a high percentage of the time that might approach 100% availability, but a small percentage of down time is acceptable and expected.

In this way, high-availability can be defined as a highly available resource that is almost always operational and accessible to clients.

The section explains the following type of high availability: "HACMP for AIX scenario - Backup domain manager”

---

**Agent availability versus job availability**

Having Tivoli Workload Scheduler working in a Windows cluster and HACMP for AIX environments does not mean that the jobs the scheduler launches are automatically aware of the cluster. It is not the responsibility of Tivoli Workload Scheduler to roll back any actions that a job might have performed during the time it was running. It is the responsibility of the user creating the script or command to allow for a roll back or recovery action in case of failover.

For a failover, the Tivoli Workload Scheduler agent reports any job running at that moment in the **ABEND** state with return code RC=0. This prevents any further dependencies being released. Only a recovery (or rerun) of the failing jobs is possible.

In general, Tivoli Workload Scheduler does not manage job and job stream interruption. Extra logic needs to be added by the user to recover job and job stream interruptions (see sections 1.4.2 and 2.3.4 of the Redbook *High Availability Scenarios with IBM Tivoli Workload Scheduler and IBM Tivoli Framework*).

---

**HACMP for AIX scenario - Backup domain manager**

Tivoli Workload Scheduler provides a degree of high-availability through its backup domain manager feature, which can also be implemented as a backup master domain manager.

The backup domain manager duplicates changes to the production plan of the domain manager. When a failure is detected, the **switchmgr** command is issued to all workstations in the domain of the domain manager server, causing the workstations to recognize the backup domain manager.
However there are cases where a cluster environment represents a suitable alternative:

- Difficulty in implementing the automatic domain responsibility switch
- Difficulty in switching jobs that should run on the domain manager to the backup domain manager
- The need to notify the switch of a domain manager to the Tivoli Workload Scheduler network
- A high-availability product addresses many of the coding issues that surround detecting hardware failures
- Implementing high-availability for fault-tolerant agents (extended agents and standard agents) cannot be accomplished using the backup domain manager feature
Integration of Tivoli Workload Scheduler and Tivoli System Automation for Multiplatforms to implement a highly available scheduling environment.

Tivoli Workload Scheduler consists of multiple built-in high-availability features that you use to switch components to other workstations in the Tivoli Workload Scheduler network. When combined with Tivoli System Automation for Multiplatforms, the recovery operations:

- Are regulated by a third-party, which has more insight into the state of all critical components in the Tivoli Workload Scheduler network.
- Are issued in an automatic fashion, while maintaining dependencies and relationships between all components.
- Require less time, compared with a manual failover of all components.

For more information about the requirements and configuration steps to implement this integration, see [Integrating with Tivoli System Automation for Multiplatforms](#).
Chapter 3. Tivoli Workload Scheduler with Windows cluster

This section contains information on the following topics:

- “Windows cluster overview”
- “Tivoli Workload Scheduler Windows Cluster Enabler” on page 7
- “Troubleshooting” on page 35

Windows cluster overview

This section describes how Windows clusters fit into the topology of Tivoli Workload Scheduler. It is divided into the following subsections:

- “Tivoli Workload Scheduler with Microsoft Windows cluster environments”
- “Prerequisite knowledge”
- “Design limitations”
- “Supported operating systems” on page 6
- “Compatibility, upgrade, and coexistence” on page 6
- “Security and Authentication” on page 7

Tivoli Workload Scheduler with Microsoft Windows cluster environments

Tivoli Workload Scheduler can be integrated into the Windows cluster environments using Microsoft generic cluster resources. This document describes how this is achieved.

To help you perform this integration, the product provides:

- A utility that remotely configures Tivoli Workload Scheduler on all the nodes of the cluster without reinstalling Tivoli Workload Scheduler on each node. The utility implements the logic to define and install the Tivoli Workload Scheduler custom resource within a cluster group.
- A new custom resource DLL specifically for Tivoli Workload Scheduler.

Prerequisite knowledge

To understand the issues discussed in this document, you must be conversant with Tivoli Workload Scheduler and Microsoft Windows clusters:

Tivoli Workload Scheduler

For an overview of Tivoli Workload Scheduler refer to the Tivoli Workload Scheduler: Planning and Installation Guide

Microsoft Windows clusters

For a Quick Start Guide for Server Clusters, and information about Windows Clustering Services go to the Microsoft Windows Server TechNet website.

Design limitations

The following design limitations apply:

- “The master domain manager” on page 6
- “Tivoli Workload Scheduler commands” on page 6
Use with multiple agents

The Tivoli Workload Scheduler master domain manager is not supported as a cluster resource for the following reasons:

- The master domain manager runs the JnextPlan critical job stream. The responsibility of the job stream is to create a new plan for the current production day. This process cannot be interrupted. An interruption might cause malfunctions and scheduling service interruptions. Only manual steps can be used to recover from such malfunctions or service interruptions. Because failover of the cluster group that contains the Tivoli Workload Scheduler resource stops the agent on the current node and starts it on a different node, if failover happens during running of JnextPlan it could be destructive.

- The Tivoli Workload Scheduler command-line utilities (conman, composer, and so on) are not aware of the cluster and if they are interrupted (through a failover of the cluster group that contains the Tivoli Workload Scheduler resource) they might corrupt some vital information for Tivoli Workload Scheduler.

Tivoli Workload Scheduler commands

Any Tivoli Workload Scheduler command that is running during a failover is not automatically taken offline (unlike the main processes netman, mailman, batchman, and jobman) by the Tivoli Workload Scheduler cluster resource.

This could be particularly problematical if the failover happens during an ad-hoc submission. The job submitted could remain in the ADDING state forever.

Use with multiple agents

If you plan to use multiple agents on the same 32-bit Windows server, you must take steps to reconfigure the Windows desktop heap memory so that the multiple agents processes share more desktop heap memory. These steps are described in "Resolving desktop heap size problems on workstations with more than three agents," on page 53.

Supported operating systems

The Tivoli Workload Scheduler Windows Cluster Enabler is available for both 32-bit and 64-bit Windows systems.

Compatibility, upgrade, and coexistence

The Tivoli Workload Scheduler agent configured to work in a cluster environment does not impact compatibility with previous Tivoli Workload Scheduler versions and does not require configuration or data migration.

A Tivoli Workload Scheduler agent configured to work in a Windows 2003 cluster environment can be connected to both the distributed and the end-to-end network configurations.

The DLL that extends the Windows Cluster Administration program is sometimes updated in fix packs and new releases of Tivoli Workload Scheduler. For this reason, the program that installs the Windows Cluster Enabler has an update option that you use to update the DLL with a new version, minor (fix pack) or major (new release of Tivoli Workload Scheduler).
Security and Authentication
The usual Tivoli Workload Scheduler security authentication and authorization mechanism applies.

Tivoli Workload Scheduler Windows Cluster Enabler
This section describes the implementation of the Windows Cluster Enabler. It consists of the following subsections:

- “Components”
- “Installation and configuration” on page 9
- “Upgrading from V8.4 or later to the general availability (GA) version manually” on page 17
- “twsClusterAdm command with examples of usage” on page 18
- “Operating Tivoli Workload Scheduler in Windows cluster environment” on page 29
- “Uninstalling Tivoli Workload Scheduler” on page 34

Components
The Tivoli Workload Scheduler Windows Cluster Enabler consists of the following elements:

- A utility to:
  - Install and remotely configure Tivoli Workload Scheduler on all the other nodes of the cluster
  - Install and configure the Tivoli Workload Scheduler cluster resource type for a given virtual server
- A Tivoli Workload Scheduler Manager Custom Resource type to manage cluster events for IBM Tivoli Workload Scheduler instances (new DLLs)
- A Tivoli Workload Scheduler extension DLL to extend the Windows Cluster Administration program
The main component is the custom resource DLL. It has the following characteristics:

- It can be brought online and taken offline
- It can be managed in a cluster
- It can be hosted (owned) by only one node at a time

As illustrated in Figure 1, the Cluster service communicates with the custom resource DLL through the resource monitor to manage resources. In response to a Cluster service request, the resource monitor calls the appropriate entry-point function in the custom resource DLL to check and control the resource state (possibly the Tivoli Workload Scheduler agent).

The custom resource DLL either performs the operation, signals the resource monitor to apply default processing (if any), or both. The custom resource DLL is responsible for providing entry-point implementations that serve the needs of the Tivoli Workload Scheduler resources.

The Tivoli Workload Scheduler Manager custom resource DLL provides the following entry-points (or services):

**IsAlive**
Determines if the Tivoli Workload Scheduler agent is currently active.

**Offline**
Performs a graceful shutdown of the Tivoli Workload Scheduler agent.

**Online**
Starts the Tivoli Workload Scheduler agent, links the agent to the network, and makes the resource available to the cluster.

**Terminate**
Performs an immediate shutdown of the resource.

---

**Figure 1. Main components of the Tivoli Workload Scheduler Cluster Enabler**

The diagram shows the main components of the Tivoli Workload Scheduler Cluster Enabler. The Cluster service and Cluster Admin Extension communicate with the custom resource DLL, which in turn interacts with the Tivoli Workload Scheduler Agent. The Resource Monitor and Cluster Administration are also involved in the process.
The Tivoli Workload Scheduler Manager custom resource DLL is a bridge between the resource monitor (part of the Windows cluster service) and the Tivoli Workload Scheduler agent. The most important objective of the custom resource DLL is to understand the agent state and to bring it online or offline using the correct sequence of commands.

**Installation and configuration**

This section describes the installation and configuration of the Windows Cluster Enabler. It is divided into the following subsections:

- “Windows Cluster Enabler”
- “Installing in a cluster”
- “Prerequisites” on page 11
- “Install and configure a new Tivoli Workload Scheduler agent” on page 12

The Tivoli Workload Scheduler Windows Cluster Enabler is installed automatically by Tivoli Workload Scheduler versions 9.1. A new folder, named `cluster`, is created within the Tivoli Workload Scheduler installation directory.

**Windows Cluster Enabler**

The Tivoli Workload Scheduler Windows Cluster Enabler includes the following files:

- **ITWSWorkstationEx.dll**
  The Tivoli Workload Scheduler Cluster Administrator extension. It adds a new property sheet and wizard pages for the Tivoli Workload Scheduler resource type to the Cluster Administrator console. See “Tivoli Workload Scheduler Cluster Administrator extension” on page 33 for more details.

- **twsClusterAdm.exe**
  Used to install and configure Tivoli Workload Scheduler.

- **ITWSResources.dll**
  The dynamic-link library containing the implementation of the Resource API for the Tivoli Workload Scheduler ITWSWorkstation resource type. It implements the logic that enables the Resource Monitor to monitor and manage the Tivoli Workload Scheduler agent

- **ITWSExInst.cmd**
  The sample script that registers the Tivoli Workload Scheduler Cluster Administrator extension.

**Installing in a cluster**

A minimal cluster configuration is composed of two nodes. Complying with the used disk technology, a Windows cluster can have from 2 to 36 nodes.
On each node run zero, one, or more cluster resource groups. In case of failure, for example, of node A all the cluster resource groups associated to the failing node failover to node B. In this way node B runs all the cluster-aware applications that were running on node A.

To have Tivoli Workload Scheduler working in a cluster environment you can:

- Install the Tivoli Workload Scheduler agent locally on the hard disk of one of the nodes if you need to schedule on that cluster node only (as a single computer). This works like a normal installation. No cluster awareness is required.
- Install the Tivoli Workload Scheduler agent on one or more virtual servers if you need to schedule jobs on that virtual server. Cluster awareness is required.

A virtual server is a group containing a network name resource, an IP address resource, and additional resources necessary to run one or more applications or...
services. Clients can use the network name to access the resources in the group, analogous to using a computer name to access the services on a physical server. However, because a virtual server is a group, it can be failed over to another node without affecting the underlying name or address.

To configure Tivoli Workload Scheduler to work in Windows cluster environment you are required to create a virtual server, adding to it a physical disk resource type and installing Tivoli Workload Scheduler on that disk.

The new cluster resource type created to manage a Tivoli Workload Scheduler agent will perform a graceful shutdown and start up the agent during a failover.

**Prerequisites**

The following are prerequisites for the correct setup of IBM Tivoli Workload Scheduler on the cluster:

**Windows Cluster Server**

A fully configured, up and running Windows Cluster Server must be ready.

**A configured Cluster Virtual Server Group for Windows 2003**

The Cluster Virtual Server Group is a group containing at least the virtual IP address resource, the network name resource, and the physical disk resource. The Cluster Virtual Server Group can contain other application resources, not only IBM Tivoli Workload Scheduler ones.

To create the Cluster Virtual Server you can use the Cluster Administrator console. You have to do the following:

- Create the Cluster Group
- Add the Shared Disk to the Cluster Group created
- Add the IP Resource and create dependency from the Disk
- Add the Network Resource and create dependency from the IP Resource

See the Windows documentation for more details.

**A configured File Server Service for Windows 2008**

The File Server Service is a service containing at least the virtual IP address resource, the network name resource, and the physical disk resource. The Cluster File Server resource can contain other application resources, not only IBM Tivoli Workload Scheduler ones.

To create the File Server Service you can use the Cluster Administrator console. You have to do the following:

- Create the File Server
- Associate the static IP address to the File Server Resource
- Add the Shared Disk to the File Server Resource created
- The system automatically adds to the File Server Resource the Network Name Resource with the same name

See the Windows documentation for more details.

**A Domain Administrator user**

A Domain Administrator user ready to use (the user should belong to the Administrator group of all the nodes of the cluster) and password.

**A domain user**

Specify a domain user as an IBM Tivoli Workload Scheduler user during the installation. If a valid domain is not specified a local user is created by default.
Grant access rights to Cluster Administrator

Verify that the cluster administrator account has the following right:

**Replace a process level token.** To add this right to the Cluster Administrator account open *Control Panel → Administrative Tools → Local Security Policy → Local Policies → User Rights Assignment* and add the Cluster Administrator user account to the **Replace a process level token** security policy list. This right is required to enable the Cluster Administrator to act as the Tivoli Workload Scheduler user. In this way the Tivoli Workload Scheduler custom resource, that runs with the rights of the Cluster Administrator user, is able to stop, start, and link Tivoli Workload Scheduler. Reboot the cluster nodes to have this change take effect. This operation is required only the first time you configure Tivoli Workload Scheduler to work in the Windows cluster environments.

Install Microsoft Visual C++ 2005 Redistributable Package (x86 or x64) on other cluster nodes

All nodes in the cluster must be able to support the use of C++. This is achieved on a given node by installing the *Microsoft Visual C++ 2005 Redistributable Package (x86 or x64)*. The installation of the Tivoli Workload Scheduler cluster enabler installs this package on the node where the enabler is installed, but to allow you to switch to the other nodes in the cluster, the package must be installed on them, too.

Follow this procedure:

2. Copy the package to each node in the Cluster Virtual Server Group:
3. On each node in the group (other than that on where you will install the cluster enabler), do the following:
   a. Log on as Domain Administrator
   b. Run vcredist_x86.exe or vcredist_x64.exe

**Install and configure a new Tivoli Workload Scheduler agent**

*Note:* During Tivoli Workload Scheduler 9.1 Cluster installation the following parameters must not be specified in double-byte character set (DBCS) characters:

- User
- Domain

To install Tivoli Workload Scheduler in a cluster-aware configuration, use the following procedure:

1. Install the Tivoli Workload Scheduler agent:
   a. Select one node of the cluster. This node must be used for any subsequent operations (such as fix pack installations).
   b. Log on to the node using a user with Domain Administrator privileges.
   c. Choose the Microsoft Virtual Server where you want to install the IBM Tivoli Workload Scheduler agent.
   d. Install Tivoli Workload Scheduler version 9.1:
2. Make the Tivoli Workload Scheduler cluster aware:
   a. Run the Windows Command Prompt.
   b. Move into the Tivoli Workload Scheduler home directory (on the shared disk).
   c. Run `tws_env.cmd` to load the Tivoli Workload Scheduler environment variables.
   d. Run `Shutdown.cmd` to stop Tivoli Workload Scheduler.
   e. Move into the cluster directory.
   f. Run the utility `twsClusterAdm.exe` to configure Tivoli Workload Scheduler remotely on all nodes of the cluster and to install the Tivoli Workload Scheduler Cluster Resource. See "Example 1: First installation of Tivoli Workload Scheduler in a Windows cluster environment" on page 25 for an installation example.

3. Define a new workstation object on the master domain manager using either composer or the Dynamic Workload Console. Verify that the node name specified is resolved by the DNS and the IP address can be pinged from the master domain manager. If you are using end-to-end network configuration, you must specify the IP Address that you specified for the NET/IP Cluster Resource Value.

4. Start the Tivoli Workload Scheduler Cluster Resource.

   **On Windows 2003 operating systems:**
   From the Cluster Administrator console select the cluster group (in Figure 4 on page 14 it is Virtual Server F).

   **On Windows 2008 operating systems:**
   From the Cluster Administrator console select the file server (in Figure 5 on page 15 it is twsFS).

5. Select the ITWSWorkstation Resource Type, right-click and select **Bring Online**:

   **On Windows 2003 operating systems:**
   In Figure 4 on page 14 the instance name is ITWSWorkstation_CLUSTER_SA_DM1.

   **On Windows 2008 operating systems:**
   In Figure 5 on page 15 the instance name is ITWSWorkstation_CLUSTER_aia86.
Figure 4. Cluster group example on Windows Server 2003
6. Wait until the Final job stream runs or generate a new plan to add the new workstation to the plan.

Upgrading agents to a new version and available fix packs

Upgrading agents to a new version and available fix packs.

To upgrade automatically Tivoli Workload Scheduler cluster nodes from V8.5.1 or V8.6 and their available fix packs to the general availability (GA) version of the product:

![Figure 5. File Server example on Windows Server 2008](image)

Use the `twsClusterUpg` command that is described in "Upgrading automatically from V8.5.1 or V8.6 and their fix packs by using `twsClusterUpg`.

To upgrade the Tivoli Workload Scheduler nodes from V8.4 or later to the general availability (GA) version of the product:

Follow the procedure that is described in "Upgrading from V8.4 or later to the general availability (GA) version manually" on page 17.

Upgrading automatically from V8.5.1 or V8.6 and their fix packs by using `twsClusterUpg`

Use the `twsClusterUpg` command to upgrade automatically Tivoli Workload Scheduler cluster nodes from V8.5.1 or V8.6 and their fix packs to the general availability (GA) version of the product.

To perform the upgrade, run the following procedure:
1. Ensure that your environment meets the prerequisites that are listed in "Prerequisites" on page 11.

2. Move to the directory where you downloaded the images of the general availability (GA) version.

3. On all the nodes you want to upgrade, run the `twsClusterUpg` script with Cluster Administrator and Domain Administrator rights by using the following syntax:
   
   ```
   cscript.exe twsClusterUpg.vbs
   -groups <group1, ..., groupn>
   -passwords <pwd1, ..., pwdn>
   ```

   Where:

   `group1, ..., groupn`
   Specify the groups that you want to upgrade.

   `pwd1, ..., pwdn`
   Specify the password of the Tivoli Workload Scheduler user you had used to install the product in the group. Specify the passwords in the same order that you specify the groups.

   The script pauses all the nodes where Tivoli Workload Scheduler is installed and proceeds to upgrade the product group by group. If errors occur, analyze the `%temp%\TWA\TWSClusterMainLog.txt` file.

   Where `%temp%` is the temporary directory of the user that is running the `twsClusterUpg` command. The `twsClusterUpg` command runs the `twstin.vbs`, `twspatch.vbs`, and the `twsClusterAdm.exe` commands. If the `TWSClusterMainLog.txt` indicates that one of these commands fails, analyze the corresponding log for further details.

4. Resume all the nodes of the cluster you upgraded, by running the following command against each node:
   ```
   cluster.exe node <node_name> /Resume
   ```

   Where `node_name` is the name of the node you want to resume.

5. Bring online the Tivoli Workload Scheduler resources on all the nodes, by running the following command against each resource:
   ```
   cluster.exe res <res_name> /Online
   ```

   Where `<res_name>` is the name of the Tivoli Workload Scheduler resource.

   For an example about how to use `twsClusterUpg`, see **twsClusterUpg example to upgrade from V8.5.1 Fix Pack 4 to V9.1**.

   **twsClusterUpg example to upgrade from V8.5.1 Fix Pack 4 to V9.1**

   This example shows the syntax that you can use to upgrade an agent from V8.5.1 Fix Pack 4 to V9.1 and the output you receive.

   In this example the group to upgrade is `twsFS`.

   Syntax
   ```
   cscript.exe twsClusterUpg.vbs -groups twsFS -passwords mypwd
   ```

   Output
Groups to upgrade:
- twsFS

AWSFAB323I Pausing node nc112087
OK
AWSFAB323I Pausing node nc112088
OK
AWSFAB344I The product is setting all the Tivoli Workload Scheduler cluster resources on the node TIVVM234 to offline.
AWSFAB336I The resource ITWSWorkstation_TWSCluster_RES was set to offline.
OK
AWSFAB345I The product is upgrading the cluster group twsFS with the resource ITWSWorkstation_TWSCluster_RES.
This operation might take some time.
AWSFAB347I The cluster group twsFS was upgraded successfully.

Groups upgraded successfully:
- twsFS

Upgrading from V8.4 or later to the general availability (GA) version manually

If you have Tivoli Workload Scheduler V8.4 or V8.5 installed, you can upgrade the agent to the general availability (GA) version of the product only by running the following procedure. If you have Tivoli Workload Scheduler V8.5.1 or later, you can upgrade the agent to the GA version by running the following procedure or by using twsClusterUpg.

1. Ensure your environment meets the prerequisites listed in "Prerequisites" on page 11.

2. Set all the nodes of the cluster to the Pause state. You must pause all the nodes in which you defined at least one cluster resource. In this way the Tivoli Workload Scheduler cluster resources cannot be moved from one node to another. With this operation you fix the resources on the node where they are running. Perform this action by running the following command against each node:
   ```cmd
   cluster.exe node <node_name> /Pause
   ```
   Where `<node_name>` is the name of the node to pause.

3. Set offline all the Tivoli Workload Scheduler resources belonging to the nodes of the cluster you paused to prevent Tivoli Workload Scheduler from upgrading the resources DLL with a cluster service that is still loading the DLL. Perform this action by running the following command against each resource:
   ```cmd
   cluster.exe res <res_name> /Offline
   ```
   Where `<res_name>` is the name of the resource to set offline.

4. For all the nodes in the cluster, upgrade all the resources running on the nodes, by performing the following steps:
   a. Generate the installation registries for one of the cluster groups belonging to the node on which you are upgrading the resources, and upgrade the instance you are working on, using one of the following script:
      ```cmd
      cscript.exe twsinst.vbs -update -uname <user_name>
      -password <TWS_user_password>
      -inst_dir TWS_install_dir -recovInstReg true
      ```
   b. Update the remote Windows Services and the resource DLL, by running the following command:
      ```cmd
      twsClusterAdm.exe -update resource=<res_name> ask=yes
      -twsupd pwd <TWS_user_password>
      ```
Where `<res_name>` is the name of the resource you are upgrading and
`<TWS_user_password>` is the Windows password of the Tivoli Workload Scheduler user.

**Note:** If you added a node to your Tivoli Workload Scheduler cluster after installing Tivoli Workload Scheduler, run the following command:
```
twsClusterAdm.exe -update hosts=<hostname1>,<hostname2>,... resource=<res_name> ask=yes -twsupd pwd <TWS_user_password>
```

Where `<hostname1>,<hostname2>,...` are the host names of the cluster nodes that you added after the installation and you want to upgrade.

c. Repeat steps 4a on page 17 and 4b on page 17 for all the resources present on this node.

5. Resume all the nodes of the cluster you paused in the Step 1 by running the following command against each node:
```
cluster.exe node <node_name> /Resume
```

Where `<node_name>` is the name of the node you want to resume.

6. Bring online the Tivoli Workload Scheduler resources on all the nodes, by running the following command against each resource:
```
cluster.exe res <res_name> /Online
```

Where `<res_name>` is the name of the Tivoli Workload Scheduler resource.

See “Example 10: Upgrade from Tivoli Workload Scheduler version 8.4 or later, cluster enabled” on page 27 for an upgrade example.

**twsClusterAdm command with examples of usage**

The `twsClusterAdm` command is the utility to configure Tivoli Workload Scheduler in the Microsoft Windows cluster environments.

The `twsClusterAdm` command:
- Configures Tivoli Workload Scheduler on all the nodes of the cluster or on a new joining cluster node.
- Installs the new Tivoli Workload Scheduler Cluster resource type for a first time installation. The name of this new cluster resource type is `ITWSWorkstation`.
- Creates an instance of the Tivoli Workload Scheduler Cluster resource type within a cluster group.
- Removes Tivoli Workload Scheduler configuration from one or more nodes of the cluster.
- Upgrades the Tivoli Workload Scheduler Cluster resource type if a new version is available.

It works in several steps to complete a cluster-aware installation:
- Determines if setup is running in a cluster environment.
- Copies the Tivoli Workload Scheduler resource DLL to the cluster nodes.
- Updates Tivoli Workload Scheduler services startup mode from automatic to manual.
- Installs Tivoli Workload Scheduler services and registry key on the other nodes (because the current services and registry key was installed by the normal product installation).
- Registers Tivoli Workload Scheduler Resource Types.
• Creates a new instance of the Tivoli Workload Scheduler resource within a given cluster group (virtual server) and register the instance name in the localopts file.

• Installs Common Inventory Technology on the other nodes

**Configuring a cluster-aware domain manager**

If you want to install a domain manager on a cluster environment, you must specify the `link` option using the `twsClusterAdm` parameter `opts` (described in “Syntax”). Specify the parent domain of the domain manager, and the parent domain manager workstation, so that the installation process can correctly manage the unlinking and relinking required. See “Example 9: First installation of domain manager in Windows cluster environment, specifying generic options” on page 27 for a worked example.

If the `link` option is omitted, or supplied with incorrect values, the configuration cannot complete correctly. However, you do not need to repeat the installation to resolve the problem. Instead, go to the Tivoli Workload Scheduler resource instance property panel, and under the **Parameters** tab add the link in the `genericOpts` field. When you activate the cluster the information in the `link` option is used to complete the configuration.

**Syntax**

```
twsClusterAdm.exe -new
  domain=<Windows_domain>
  user=<TWS_user>
  pwd=<TWS_user_password>
  [hosts=<hostname1,hostname2...>]
  [twshome=<TWS_home_dir>]
  [ –res
group=<cluster_group_name>
ip=<IP_resource_name>
net=<network_resource_name>
disk=<disk_resource_name>
[rename=<resource_instance_name>]
[check_interval=<TWS_polling_interval>]
[failover=yes | no]
[looksalive=<lookalive_interval>]
isalive=<isalive_interval>]
[tcpport=<tcp_port>]
[opts=<generic_options>]
]
[–notwsinst]
[–dll
  [path=<DLL_resource_path>]]
[–force]
[–sharedDesktop [name=<desktop_name>]]
twsClusterAdm.exe –uninst
  domain=<Windows_domain>
  user=<TWS_user_name>
  [hosts=<hostname1,hostname2...>]
twsClusterAdm.exe –update
  resource=<resource_instance_name>
```
twsClusterAdm.exe –changeResName
"<resource_instance_name>"
"<new_resource_instance_name>"

Parameters and arguments

–new  The –new parameter configures Tivoli Workload Scheduler on all the nodes of the cluster or on a new cluster node. It takes the following arguments:

Domain=<Windows_domain>
The Windows User Domain of the Tivoli Workload Scheduler User.
This parameter is mandatory if –new or –uninst is specified. This parameter must not be specified in double-byte character set (DBCS) characters.

user=<TWS_user>
The Windows User Name of the Tivoli Workload Scheduler User.
This parameter is mandatory if –new or –uninst is specified. This parameter must not be specified in double-byte character set (DBCS) characters.

pwd=<TWS_user_password>
The Windows password of the Tivoli Workload Scheduler user.
This parameter is mandatory if –new is specified.

hosts=<hostname1,hostname2...>
The host names of the cluster nodes where you want to configure Tivoli Workload Scheduler. Host names must be separated by commas. This parameter is optional. It can be used to configure a new joining node of the cluster.

twshome=<TWS_home_directory>
The directory where Tivoli Workload Scheduler is installed. This parameter is optional. If you do not specify this directory, the command will discover the installation directory.

–res  The –res parameter adds a new instance of the Tivoli Workload Scheduler resource type to an existing cluster group. It takes the following arguments:

group=<cluster_group_name>
The name of the group (Virtual Server) where Tivoli Workload Scheduler is configured as the cluster resource. This parameter is mandatory.

ip=<IP_resource_name>
The name of the cluster IP resource type that the Tivoli Workload Scheduler resource depends on. This parameter is mandatory.

net=<network_resource_name>
The name of the network resource type that the Tivoli Workload Scheduler resource depends on. This parameter is mandatory.

disk=<disk_resource_name>
The name of the disk resource type that the IBM Tivoli Workload Scheduler resource depends on. This parameter is mandatory.
resname=<resource_instance_name>
The name of the resource instance, as it appears in the Cluster Administrator (see Figure 4 on page 14). If this parameter is not supplied, the default value of ITWSWorkstation_<domain_name>_<user_name> is used.

failover=yes|no
If you specify yes, Tivoli Workload Scheduler can cause the failover of the virtual server group. If you do not specify this option Tivoli Workload Scheduler will not cause the failover of the virtual server group. This parameter is optional. Note that you can modify this setting directly from the Cluster Administrator console. Modifying the threshold and period values from the resource property tab you can enable or disable the automatic failover in case of resource failure. See the Windows Cluster Guide for more information.

check_interval=<TWS_polling_interval>
The interval in milliseconds that the Tivoli Workload Scheduler resource waits between two health checks. This parameter is optional. Use values greater than 60000. The default value is 100000. You can change this value from the Cluster Administrator console: right-click the resource and select Properties → Parameters.

lookalive=<isalive_interval>
The interval in milliseconds at which the Cluster service polls the resource to determine if it appears operational. This parameter is optional. Use values greater than 10000. The default value is 10000. You can change this value from the Cluster Administrator console: right-click the resource and select Properties → Advanced.

Isalive=<isalive_interval>
The interval in milliseconds at which the Cluster service polls the resource to determine if it is operational. This parameter is optional. Use values greater than 10000. The default value is 60000. You can change this value from the Cluster Administrator console: right-click the resource and select Properties → Advanced.

tcpport=tcp_port
This parameter is reserved for future use.

opts=generic_options
The generic resource options is used to specify a set of options. Each option is separated by a semicolon ";". The opts parameter will accept the following options:

• ftaOff
  Use this option to enable monitoring of the dynamic scheduling agent. If not specified, the default behavior remains unchanged: only fault tolerant agent is monitored.

• lwaOn
  Use this option to enable monitoring of the dynamic scheduling agent and Job Manager. If not specified, the default behavior remains unchanged: only fault tolerant agent is monitored.

• killjob
  Use this option to kill any job (and job child) running at the moment of the resource failure.

• link=<parent_domain>!<parent_domain_manager>
Use this option if you are configuring a cluster-aware domain manager. Specify the parent domain and manager of the agent you are configuring.

For example if you are configuring a domain manager which is a child of the master domain manager (named MyMasterWS in the domain MASTERDOM), the value to specify is link=MASTERDOM!MyMasterWS.

The **kill** and the **link** options can be used together (for example, opts=kill;link=MASTERDOM!MASTERWS). You can change this value from the Cluster Administrator console: right-click the resource and select **Properties > Parameters**. Change the values in the **genericOpts** field.

### –notwsinst

The **–notwsinst** parameter optionally specifies that you are installing the cluster resource instance for an existing instance of Tivoli Workload Scheduler.

### –dll

The **–dll** parameter specifies that the ITWSResources.dll that implements the new Tivoli Workload Scheduler resource type needs to be installed. This parameter is mandatory the first time you configure Tivoli Workload Scheduler on the cluster or if a new node is joining the cluster. This parameter takes one optional argument:

```
[path=<DLL_resource_path>]
```

The path where the ITWSResources.dll must be installed. This parameter is optional. If you do not specify the path, the default value, \%systemRoot\%\cluster, is used. **Do not** specify the drive letter for the path. The path specified must exist and must be accessible on each node of the cluster.

### –force

The **–force** parameter optionally forces the installation of the Tivoli Workload Scheduler resource DLL (ITWSResources.dll) without checking the version. The parameter is ignored if you did not specify the **–dll** parameter.

### –sharedDesktop

The **–sharedDesktop** parameter optionally specifies that **Jobmon** uses a shared desktop name to manage desktop heap memory allocation where multiple agents are installed on one computer (see **Resolving desktop heap size problems on workstations with more than three agents** on page [53](#) for details). Use the same name for at least two agents on this computer to make the option effective.

```
name=<desktop_name>
```

The optional desktop name. If you supply a name, it must be in single-byte characters (English alphabet), with no special characters allowed, except spaces, in which case you must surround it by double quotes. The default name (by not supplying the **name=<desktop_name>** argument), is TWS_JOBS_WINSTA.

### –uninst

The **–uninst** parameter uninstalls the cluster resource instance, and accepts the following arguments:

```
Domain=<Windows_domain>
```

The Windows User Domain of the Tivoli Workload Scheduler User.

---

22 Tivoli Workload Scheduler: High Availability Cluster Environments
This parameter is mandatory if you specify –new or –uninst. This parameter must not be specified in double-byte character set (DBCS) characters.

user=<TWS_user>
The Windows User Name of the Tivoli Workload Scheduler User. This parameter is mandatory if you specify –new or –uninst. This parameter must not be specified in double-byte character set (DBCS) characters.

hosts=<hostname1,hostname2...>
The host names of the cluster nodes where you want to uninstall Tivoli Workload Scheduler. Host names have to be separated by commas. This parameter is optional. If you do not specify this parameter, Tivoli Workload Scheduler is unloaded from all nodes in the cluster except for the current node.

–update
The –update parameter updates the Tivoli Workload Scheduler resource DLL of an existing instance, and accepts the following arguments:

resource=<resource_instance_name>
The Tivoli Workload Scheduler Resource Instance name as it appears within the cluster group. The default name is ITWSWorkstation_<domain>_<user>. This parameter is mandatory.

ask={yes | no}
Define if the updater should ask before restarting the resource DLL after updating it. Supply yes to determine that the updater should stop and ask the operator to confirm that it can restart the cluster resource DLL. The default value is yes. Supply no to automatically restart the cluster resource DLL after upgrading it, without a manual intervention. This parameter is optional.

–force
Force the installation of the Tivoli Workload Scheduler resource DLL (ITWSResources.dll) without checking the version. This parameter is optional.

–twsupd
Define whether to update the Windows service registry after updating the resource DLL. Use this parameter only when updating the DLL after upgrading Tivoli Workload Scheduler to a new major version, such as 9.1. This parameter is optional.

hosts=<hostname1,hostname2...>
The host names of the cluster nodes on which you want to update Tivoli Workload Scheduler. Host names have to be separated by commas. This parameter is optional. If you do not specify this parameter, Tivoli Workload Scheduler is updated on all nodes in the cluster.

–changeResName
The –changeResName parameter changes the cluster instance resource name, and accepts the following arguments:

"<resource_instance_names>"
The Tivoli Workload Scheduler Resource Instance name as it appears within the cluster group. The default name is ITWSWorkstation_<domain>_<user>. This argument is mandatory for –changeResName.
The new name you want to use for the Tivoli Workload Scheduler resource instance. This argument is also mandatory for `--changeResName`.

**Examples**

For all the examples described below it is assumed that Tivoli Workload Scheduler 9.1 has been installed.

In all the examples described below the following definitions are used:

- **mydom**  
  Is the Windows User Domain of the Tivoli Workload Scheduler user.

- **mytwsuser**  
  Is the Tivoli Workload Scheduler user name.

- **mytwspwd**  
  Is the password for the `MYDOM\mytwsuser` domain user.

- **myresgroup**  
  Is the name of the cluster resource group selected.

- **myip**  
  Is the name of the IP Address resource type within the `myresgroup` resource group.

- **mynetname**  
  Is the name of the Network Name resource type within the `myresgroup` resource group.

- **mysharedDesktop**  
  Is the name of the shared desktop that all instances of `jobmon` will use.

- **myResName**  
  Is the customized name of the resource instance.

The examples are as follows:

- **Example 1**: First installation of Tivoli Workload Scheduler in a Windows cluster environment on page 25
- **Example 2**: Install and configure the new custom resource for an existing installation of Tivoli Workload Scheduler on page 25
- **Example 3**: Add a new agent in a cluster environment with Tivoli Workload Scheduler already installed on page 25
- **Example 4**: Add a custom resource type instance to an existing cluster group on page 26
- **Example 5**: Configure Tivoli Workload Scheduler in a new joining node of the cluster on page 26
- **Example 6**: Deregister Tivoli Workload Scheduler on all nodes of the cluster except for the current node on page 26
- **Example 7**: Install a new version of the cluster resource DLL into the cluster on page 26
- **Example 8**: Force the upgrading/downgrading of the cluster resource DLL into the cluster on page 27
- **Example 9**: First installation of domain manager in Windows cluster environment, specifying generic options on page 27
Example 1: First installation of Tivoli Workload Scheduler in a Windows cluster environment:
First time installation of Tivoli Workload Scheduler in a Windows cluster environment.

twsClusterAdm.exe –new domain=MYDOM user=mytwsuser pwd=mytwspwd
–res group=myresgroup ip=myip net=mynetname disk=mydisk –dll

The command:
• Configures Tivoli Workload Scheduler on all the nodes of the cluster.
• Installs the new Tivoli Workload Scheduler Cluster resource type (named ITWSWorkstation) on all the nodes of the cluster.
• Copies the ITWSResources.dll to the %systemRoot%\cluster folder.
• Creates an instance of the Tivoli Workload Scheduler Cluster resource type within the specified cluster group.
• Adds a dependency from myip, mynetname, and mydisk to the resource.

Example 2: Install and configure the new custom resource for an existing installation of Tivoli Workload Scheduler:
Install and configure the new Tivoli Workload Scheduler custom resource for an existing installation of Tivoli Workload Scheduler.

Note: This example is applicable only if the Tivoli Workload Scheduler agent has already been installed for the same instance on all the nodes of the cluster and all the Tivoli Workload Scheduler services startup types are set to Manual.

twsClusterAdm.exe –new domain=MYDOM user=mytwsuser pwd=mytwspwd
–res group=myresgroup ip=myip net=mynetname disk=mydisk –dll –notwsinst

The command:
• Installs the new Tivoli Workload Scheduler Cluster resource type (named ITWSWorkstation) on all the nodes of the cluster.
• Copies the ITWSResources.dll file to the %systemRoot%\cluster folder.
• Creates an instance of the Tivoli Workload Scheduler Cluster resource type within the specified cluster group.
• Adds a dependency from myip, mynetname, and mydisk to the resource.

Example 3: Add a new agent in a cluster environment with Tivoli Workload Scheduler already installed:
Add a new Tivoli Workload Scheduler agent in a cluster environment where an agent of Tivoli Workload Scheduler has been installed and configured in a different Virtual Server.

twsClusterAdm.exe –new domain=MYDOM user=mytwsuser pwd=mytwspwd
–res group=myresgroup ip=myip net=mynetname disk=mydisk

The command:
• Configures Tivoli Workload Scheduler on all the nodes of the cluster.
• Creates an instance of the Tivoli Workload Scheduler Cluster resource type within the specified cluster group
• Adds a dependency from myip, mynetname, and mydisk to the resource.

Example 4: Add a custom resource type instance to an existing cluster group:
Add an instance of the Tivoli Workload Scheduler custom resource type to an existing cluster group.

**Note:** This example is applicable only if the Tivoli Workload Scheduler agent has been installed and configured, and the Tivoli Workload Scheduler custom resource type has been installed and registered.

twsClusterAdm.exe –new domain=MYDOM user=mytwsuser pwd=mytwpwd
–res group=myresgroup ip=myip net=mynetname disk=mydisk –notwsinst

The command:
• Creates an instance of the Tivoli Workload Scheduler Cluster resource type within the specified cluster group
• Adds a dependency from myip, mynetname, and mydisk to the resource.

Example 5: Configure Tivoli Workload Scheduler in a new joining node of the cluster:
Configure Tivoli Workload Scheduler in a new joining node of the cluster.

**Note:** This example is applicable only if the Tivoli Workload Scheduler agent has been installed and configured in the cluster environment. Possibly you have been using Tivoli Workload Scheduler for a long time, you have bought a new node for this cluster, and you want Tivoli Workload Scheduler to be able to move there in case of failure.

twsClusterAdm.exe –new domain=MYDOM user=mytwsuser pwd=mytwpwd
hosts=my_new_joining_host_name -dll

The command configures Tivoli Workload Scheduler and installs the Tivoli Workload Scheduler cluster resource DLL on the my_new_joining_host_name node.

Example 6: Deregister Tivoli Workload Scheduler on all nodes of the cluster except for the current node:
Deregister Tivoli Workload Scheduler on all the nodes of the cluster except for the current node. See the section relative to uninstall procedure for more details.

twsClusterAdm.exe –uninst domain=MYDOM user=mytwsuser

The command removes Tivoli Workload Scheduler configuration from all the nodes of the cluster except for the current node.

To uninstall Tivoli Workload Scheduler from the current node you have to use the normal uninstall procedure described in the *IBM Tivoli Workload Scheduler Planning and Installation Guide*.

Example 7: Install a new version of the cluster resource DLL into the cluster:
Install a new version of the Tivoli Workload Scheduler cluster resource DLL into the cluster.

twsClusterAdm.exe –update resource=<resource_name>

The command upgrades the Tivoli Workload Scheduler Cluster resource type if a new version is available.
Example 8: Force the upgrading/downgrading of the cluster resource DLL into the cluster:

```
twsClusterAdm.exe –update resource=<resource name> –force
```

The command upgrades the Tivoli Workload Scheduler cluster resource without verifying if the version is greater then the version of the installed version.

Example 9: First installation of domain manager in Windows cluster environment, specifying generic options:

```
twsClusterAdm.exe –new domain=MYDOM user=mytwsuser pwd=mytwspwd –res group=myresgroup ip=myip net=mynetname disk=mydisk opts=killjob;link=MASTERDM!MASTER; –dll
```

The command:

- Configures Tivoli Workload Scheduler on all the nodes of the cluster.
- Installs the new Tivoli Workload Scheduler Cluster resource type (named ITWSWorkstation) on all the nodes of the cluster.
- Copies the ITWSResources.dll to the \%systemRoot\%\cluster folder.
- Creates an instance of the Tivoli Workload Scheduler Cluster resource type within the specified cluster group.
- Adds a dependency from myip, mynetname, and mydisk to the resource.
- Sets the generic options kill and link.

Example 10: Upgrade from Tivoli Workload Scheduler version 8.4 or later, cluster enabled:

Upgrade from Tivoli Workload Scheduler version 8.4 or later, cluster enabled, to version 9.1, as follows:

1. Set all the nodes of the cluster to the Pause state. Perform this action by running the following command against each node:

   ```
   cluster.exe node NODE1 /Pause
   ```

2. Set offline all the Tivoli Workload Scheduler resources belonging to the nodes of the cluster you paused:

   ```
   cluster.exe res RES1 /Offline
   ```

3. For all the nodes in the cluster, upgrade all the resources running on the nodes, by performing the following steps:

   a. Generate the installation registries for one of the cluster groups belonging to the node on which you are upgrading the resources, and upgrade the instance you are working on, using the following script:

   ```
   cscript.exe twsinst.vbs -update -uname user1 -password us1pass1 -inst_dir "C:\Program Files\IBM\TWA" -recovInstReg true
   ```

   b. Update the remote Windows Services and the resource DLL, by running the following command:

   ```
   twsClusterAdm.exe -update resource=RES1 ask=yes -twsupd pwd us1pass1
   ```

   c. Repeat steps 3a and 3b for all the resources present on this node.

4. Resume all the nodes of the cluster you paused in step 1 by running the following command against each node:

   ```
   cluster.exe node NODE1 /Resume
   ```
5. Bring online the Tivoli Workload Scheduler resources on all the nodes, by running the following command against each resource:

```
cluster.exe res RES1 /Online
```

**Example 11: First installation of Tivoli Workload Scheduler in a Windows cluster environment, defining shared desktop:**

First time installation of Tivoli Workload Scheduler in a Windows cluster environment, defining a shared desktop to be used by **Jobmon** (this is like example 1, but with the addition of the shared desktop):

```
twsClusterAdm.exe –new domain=MYDOM user=mytwsuser pwd=mytwspwd
–res group=myresgroup ip=myip net=mynetname disk=mydisk –dll –sharedesktop
```

The command:
- Configures Tivoli Workload Scheduler on all the nodes of the cluster.
- Installs the new Tivoli Workload Scheduler Cluster resource type (named ITWSWorkstation) on all the nodes of the cluster.
- Copies the ITWSResources.dll to the %systemRoot%\cluster folder.
- Creates an instance of the Tivoli Workload Scheduler Cluster resource type within the specified cluster group.
- Adds a dependency from myip, mynetname, and mydisk to the resource.
- Defines that **jobmon** uses the default shared desktop name.

**Example 12: First installation of Tivoli Workload Scheduler in Windows cluster environment, using customized resource instance name:**

First time installation of Tivoli Workload Scheduler in a Windows cluster environment, using a customized resource instance name (this is like example 1, but with the addition of the customized resource instance name):

```
twsClusterAdm.exe –new domain=MYDOM user=mytwsuser pwd=mytwspwd
–res group=myresgroup ip=myip net=mynetname disk=mydisk resname=myResName –dll
```

The command:
- Configures Tivoli Workload Scheduler on all the nodes of the cluster.
- Installs the new Tivoli Workload Scheduler Cluster resource type (named ITWSWorkstation) on all the nodes of the cluster.
- Copies the ITWSResources.dll to the %systemRoot%\cluster folder.
- Creates an instance of the Tivoli Workload Scheduler Cluster resource type within the specified cluster group.
- Adds a dependency from myip, mynetname, and mydisk to the resource.
- Defines that the resource instance name is *myResName*.

**Example 13: Changing the resource instance name:**

Changing the name of an existing resource instance:

```
twsClusterAdm.exe –changeResName "ITWSWorkstation_CLUSTER_SA_DM1" "myResName"
```

The command changes the resource instance name from *ITWSWorkstation_CLUSTER_SA_DM1* to *myResName*.

**Example 14: First installation of domain manager in Windows cluster environment, specifying monitoring options of dynamic scheduling:**

First time installation of a Tivoli Workload Scheduler domain manager in a Windows cluster environment, specifying the fta0ff generic option.
twsClusterAdm.exe –new domain=MYDOM user=mytwuser pwd=mytwspwd –res
group=myresgroup ip=myip net=mynetname disk=mydisk
opts=ftaOff

The command:
• Configures Tivoli Workload Scheduler on all the nodes of the cluster.
• Installs the new Tivoli Workload Scheduler Cluster resource type (named ITWSWorkstation) on all the nodes of the cluster.
• Copies the ITWSResources.dll to the %systemRoot%\cluster folder.
• Creates an instance of the Tivoli Workload Scheduler Cluster resource type within the specified cluster group.
• Adds a dependency from myip, mynetname, and mydisk to the resource.
• Sets the generic option ftaOff to enable monitoring of the dynamic scheduling agent.

Operating Tivoli Workload Scheduler in Windows cluster environment

This section describes how to operate Tivoli Workload Scheduler in the Windows cluster environments. It is divided into the following subsections:

• “Cluster resource dependencies”
• “Start up and shut down Tivoli Workload Scheduler” on page 31
• “The new “cluster instance name” local option” on page 31

Cluster resource dependencies

One of the most important steps when running Tivoli Workload Scheduler in the Windows cluster environments is to verify that the dependencies have been set correctly.

To ensure Tivoli Workload Scheduler works correctly, the Tivoli Workload Scheduler cluster resource instance has to depend on the following resource types:
• IP Address
• Physical Disk
• Network Name

as shown in the following examples on Windows Server 2003 and Windows Server 2008
You can decide to add more dependencies to ensure Tivoli Workload Scheduler launches jobs only after a given service is available. This happens when Tivoli Workload Scheduler schedules jobs that prerequisite other cluster aware applications. For example, to ensure the SQL job is launched only after the cluster...
aware relational database is available, add a dependency from the relational
database cluster resource to the Tivoli Workload Scheduler cluster resource.

Start up and shut down Tivoli Workload Scheduler

The following methods can no longer be used to stop Tivoli Workload Scheduler
because they will cause a failure of the Tivoli Workload Scheduler cluster resource:

- Conman shut
- Shutdown.cmd
- StartUp.cmd
- Conman start if the ITWSWorkstation resource is offline.
- StartupLwa.cmd

Use the following scripts to stop and start Tivoli Workload Scheduler (you can
rename them if required):

- ShutDown.clu.cmd
- StartUp.clu.cmd
- ShutdownLwa.cmd

The above scripts will be automatically created under the Tivoli Workload
Scheduler installation directory by the twsClusterAdm.exe program.

If you do not use these scripts, you must run the following commands to stop and
start Tivoli Workload Scheduler services.

Stop:

```
cluster res <TWS resource instance name> /offline
```

Start:

```
cluster res <TWS resource instance name>/online
```

Examples:

If ITWSWorkstation_DOMAIN_MST_UserR is the name of the TWS resource instance, to
shut down Tivoli Workload Scheduler you have to use:

```
cluster res ITWSWorkstation_DOMAIN_MST_UserR /offline
```

To start Tivoli Workload Scheduler services you have to use:

```
cluster res ITWSWorkstation_DOMAIN_MST_UserR /online
```

where cluster is the Windows command to administer the cluster (run from the
Windows Command prompt).

The new “cluster instance name” local option

One of the steps of the twsClusterAdmin utility is instance name registration of the
Tivoli Workload Scheduler cluster resource within the local option localopts file.

The Tivoli Workload Scheduler agent uses the value of this new local option to
signal to the Tivoli Workload Scheduler cluster resource that the agent has received
a stop command.

It is important to change the value of the cluster instance name local option
every time the Tivoli Workload Scheduler resource instance name is changed. If the
cluster instance name local option does not point to the right name, the Tivoli
Workload Scheduler resource will be set to failure state from the cluster resource
monitor. Do not specify this name in double-byte character set (DBCS) characters.
To change the Tivoli Workload Scheduler resource instance name use the following procedure:

1. Take the Tivoli Workload Scheduler resource instance offline using the Cluster Administrator console. To verify if Tivoli Workload Scheduler stopped correctly you can use the Cluster Administrator console and check the status of the Tivoli Workload Scheduler resource instance. If the resource has failed to stop you can check in the cluster and Tivoli Workload Scheduler logs for the reason. See "Traces" on page 35 for more details on the log files.

2. Modify the name of the Tivoli Workload Scheduler cluster resource directly from the Cluster Administrator console, as shown in the following example on Windows Server 2003. Do not specify this name in double-byte character set (DBCS) characters.

3. Open the localopts file using Notepad. Modify the value of the cluster instance name local option. Check that the name is the same you specified for the Tivoli Workload Scheduler cluster resource in Step 2.
4. Modify the cluster instance name in the StartUp_clu.cmd and ShutDown_clu.cmd scripts.

5. Bring the Tivoli Workload Scheduler resource instance online in the Cluster Administrator console.

**Tivoli Workload Scheduler Cluster Administrator extension**

This section describes the Cluster Administrator extension. It is divided into the following subsections:

- "Cluster Administrator extension overview"
- "Installing the Cluster Administrator extension" on page 34

**Cluster Administrator extension overview**

The Cluster Administrator is a system utility with a graphical user interface that allows administrators to manage cluster objects, handle maintenance, and monitor cluster activity.

The Tivoli Workload Scheduler Cluster Administrator extension is a dynamic-link library that, when installed, extends the Cluster Administrator console with a new property sheet and a wizard page that allows you to view and edit the Tivoli Workload Scheduler resource properties.

---

*Figure 9. Localopts – Notepad*

---

<table>
<thead>
<tr>
<th>SSL Encryption Cipher</th>
<th>SSLv3</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSL auth mode</td>
<td>eonly</td>
</tr>
<tr>
<td>SSL auth string</td>
<td>tws</td>
</tr>
<tr>
<td>clusterinstance</td>
<td>=ITWSWorkstation_CLUSTER_SA_DM1_NEW_NAME</td>
</tr>
</tbody>
</table>

---

Chapter 3. Tivoli Workload Scheduler with Windows cluster  33
The graphic shows the new properties page **Parameters** tab that allows you to modify the ITWSWorkstation cluster resource parameters.

**Installing the Cluster Administrator extension**

Install this component only if you want to edit the properties directly from Cluster Administrator console.

If you do not install the Cluster Administration extension, the **Parameters** tab (see Figure 10) is not available. To modify ITWSWorkstation cluster resource parameters, you will have to change these properties using the cluster.exe system utility.

Install the Cluster Administrator extension on any computer where the Cluster Administrator console will be used.

Use the following procedure to install a new Cluster Administrator extension:

1. Copy the ITWSWorkstationEx.dll and the ITWSExInst.cmd files from the `<TWS_HOME>`\cluster directory into the directory where you want to install the Administrator Extension. You can use the default directory for the cluster: `\%systemRoot%\cluster`.
2. Double click on ITWSExInst.cmd, or run it from a command shell to install the Administrator Extension.

**Uninstalling Tivoli Workload Scheduler**

The steps to remove the product have to be launched from the same node previously used to install Tivoli Workload Scheduler and subsequent fix packs.

The procedure is as follows:

1. Manually remove the Tivoli Workload Scheduler custom resource instance from the cluster group. You can use the Cluster Administrator console to do this.
2. Optionally deregister the resource type using the command:
cluster restype ITWSWorkstation /delete

Do not deregister the resource if instances are present in other cluster groups.

3. Optionally delete the DLL ITWSResources.dll from the installation directory (the default directory is %systemRoot%\cluster).

4. Run the utility TwsClusterAdm –uninst. This utility removes the Tivoli Workload Scheduler services and registry keys from cluster nodes other than the current node. Optionally, use -host hostname1, hostname2 arguments to specify the remote nodes from which Tivoli Workload Scheduler service must be removed. Do not specify the primary node where Tivoli Workload Scheduler fault tolerant agent was installed, otherwise you will remove the original service making the Tivoli Workload Scheduler fault tolerant agent unusable.

To remove Tivoli Workload Scheduler from the current node, you can run the normal uninstall program (refer to the IBM Tivoli Workload Scheduler Planning and Installation Guide).

Troubleshooting

This part of the guide gives troubleshooting information about Tivoli Workload Scheduler in a Windows cluster environment. The information here applies to the Tivoli Workload Scheduler engine and its installation for this environment. For more troubleshooting information about Tivoli Workload Scheduler, refer to the IBM Tivoli Workload Scheduler Administration and Troubleshooting.

This chapter contains the following sections:

- “Traces” on page 36
- “Error 1314 taking online the resource and the Workstation does not link” on page 36
- “Tivoli Workload Scheduler resource instance reports fail status or Tivoli Workload Scheduler user jobs go in the abend state” on page 36
- “Windows Report panel with Jobmon.exe” on page 37
- “Cluster: IP validation error on Netman stdlist” on page 37

Traces

Tivoli Workload Scheduler maintains logs for different activities in different places. See IBM Tivoli Workload Scheduler Administration and Troubleshooting for more information.

The new cluster enablement pack introduces two trace files in the TWSInstallationDirectory\stdlist\traces directory:

clu_offline.log

When the Tivoli Workload Scheduler custom resource is taken offline (each time a failover happens), the Tivoli Workload Scheduler custom resource launches the conman command line to stop and unlink the instance. In this log you can find the output of the command.

clu_online.log

When the Tivoli Workload Scheduler custom resource is brought online (each time a failover happens), the Tivoli Workload Scheduler custom resource launches the conman command line to link the workstation to its domain manager. In this log you can find the output of the command conman link @!@;noask.
Any action the Tivoli Workload Scheduler custom resource follows is logged within the system cluster log file. This is a file named `cluster.log` located under the `%systemRoot%\cluster` folder.

### Error 1314 taking online the resource and the Workstation does not link

The problem could be related to the rights of the cluster administrator. To check this:

1. Open the cluster log file (`cluster.log`) located in the `%systemRoot%\cluster` folder.
2. Look for the strings containing `ITWSWorkstation`. These are the messages logged by Tivoli Workload Scheduler custom resource.
3. If you see a message like:
   
   ```
   <time> ERR ITWSWorkstation <resource instance name>: SubmitTwsCommand: CreateProcessWithLogonW failed <TWS_HOME>\conman.exe> < start;noask> '1314'
   ```
   
   It means that the system error 1314, A required privilege is not held by the client, occurred launching the `conman` command.
4. To solve the problem, you must give the cluster user sufficient privileges to allow custom resource instance to submit Tivoli Workload Scheduler command link.

To solve this problem, add the **Replace a process level token** right to the cluster administrator account (this is the name of the user you chose when you configured the cluster). To add this right to the Cluster Administrator account open **Control Panel → Administrative Tools → Local Security Policy → User Rights Assignment** and add the Cluster Administrator user account to the **Replace a process level token** security policy list. This right is required in order to enable the Cluster Administrator to act as the Tivoli Workload Scheduler user. In this way the Tivoli Workload Scheduler custom resource, that runs with the rights of the Cluster Administrator user, is able to stop, start, and link Tivoli Workload Scheduler. Reboot the cluster nodes to have this change take effect. This operation is required only the first time you configure Tivoli Workload Scheduler to work in the Windows 2003 cluster environment.

You must reboot the cluster nodes for this change to take effect.

### Tivoli Workload Scheduler resource instance reports fail status or Tivoli Workload Scheduler user jobs go in the abend state

**Problem:** If you run more than three instances of Tivoli Workload Scheduler on the same node with jobs running it is possible to have the following behavior:

- The Tivoli Workload Scheduler cluster resource instance is in **fail** status. See the resource status on the Cluster Administrator console (Figure 8 on page 32).
- Tivoli Workload Scheduler user jobs go in the **abend** or **fail** state. In this case you can find the following error message in `<TWS_HOME>\stdlist\date\TWSUSERNAME`:

```
+ AWSBIJ139E An internal error has occurred. Jobmon was unable to create a new desktop on the window station associated with the calling process.
+ The error occurred in the following source code file:
+ ../../src/jobmon/monutil.c at line: 2454. The error mess
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Tivoli Workload Scheduler: High Availability Cluster Environments
AWSBIJ140E An internal error has occurred. Jobmon was unable to create the Windows process environment to launch jobs. The error occurred in the following source code file: ../../src/jobmon/monutil.c at line: 830.

The following error message is in the <TWS_HOME>\stdlist\logs\date_TWSMERGE.log file:
06:00:28 19.05.2006|BATCHMAN:* AWSBHT061E Batchman has received a mailbox record indicating that the following job has terminated unexpectedly: The system has run out of desktop heap.

06:00:28 19.05.2006|BATCHMAN:* AWSBHT061E Batchman as received a mailbox record indicating that the following job has terminated unexpectedly:
   The system has run out of desktop heap.

**Solution:** The solution to this problem has a number of different options, and is described in “Resolving desktop heap size problems on workstations with more than three agents,” on page 53.

---

**Windows Report panel with Jobmon.exe**

**Problem:** After failover from node A to node B, sometimes Jobmon cause a core dump with a segmentation violation error to occur on node A. You can see the segmentation after node A is rebooted, or when logging on with Tivoli Workload Scheduler user. This does not cause a problem because Tivoli Workload Scheduler on node B works correctly after a second failover, and Tivoli Workload Scheduler also works on node A.

**Cluster: IP validation error on Netman stdlist**

This problem occurs when the node field in a workstation definition is defined as a real IP address instead of a cluster network name resource.

The problem is that the IP validation is performed using the IP address of the node when Tivoli Workload Scheduler is starting and not the IP address of the fta resources in the cluster.

You could see this warning when the parent/child agent is installed on a cluster and there is a mismatch between the real IP address that has been detected from the TCP/IP channel and the IP address declared in the definition of the workstation (property node). If the property node is a host name, this will be resolved first (querying the DNS).
Chapter 4. IBM Tivoli Workload Scheduler with HACMP

This chapter contains the following topics:
- “High-Availability Cluster Multi-Processing”
- “UNIX cluster overview” on page 42

High-Availability Cluster Multi-Processing

IBM uses the High-Availability Cluster Multi-Processing (HACMP) tool for building UNIX-based, mission-critical computing operating systems. HACMP ensures that critical resources, such as applications, are available for processing. HACMP has two major components: high availability (HA) and cluster multi-processing (CMP).

The primary reason to create HACMP clusters is to provide a highly available environment for mission-critical applications. For example, an HACMP cluster might run a database server program to service client applications. Clients send queries to the server program, which responds to their requests by accessing a database stored on a shared external disk.

In an HACMP cluster, to ensure the availability of these applications, the applications are put under HACMP control. HACMP ensures that the applications remain available to client processes even if a component in a cluster fails. To ensure availability, in case of a component failure, HACMP moves the application (together with resources needed to access the application) to another node in the cluster.

You can find more details on the following topics:
- “Benefits”
- “Physical components of an HACMP cluster” on page 40

Benefits

HACMP™ provides the following benefits:
- The HACMP planning process and documentation include tips and advice about the best practices for installing and maintaining a highly available HACMP cluster.
- When the cluster is operational, HACMP provides automated monitoring and recovery of all the resources that the application needs.
- HACMP provides a full set of tools for maintaining the cluster and ensures that the application is available to clients.

Use HACMP to:
- Set up an HACMP environment using online planning worksheets that simplify initial planning and setup.
- Ensure high availability of applications by eliminating single points of failure in an HACMP environment.
- Use high-availability features available in AIX®.
- Manage how a cluster handles component failures.
- Secure cluster communications.
• Set up fast disk takeover for volume groups managed by the Logical Volume Manager (LVM).
• Manage event processing for an HACMP environment.
• Monitor HACMP components and diagnose problems that might occur.

Physical components of an HACMP cluster

HACMP provides a highly-available environment by identifying a set of resources that are essential to uninterrupted processing, and by defining a protocol that nodes use to collaborate to ensure that these resources are available. HACMP extends the clustering model by defining relationships among cooperating processors where one processor provides the service offered by a peer, when the peer is unable to do so.

An HACMP Cluster is made up of the following physical components:

• “Nodes” on page 41
• “Shared external disk devices” on page 42
• “Networks” on page 42
• “Clients” on page 42

HACMP allows you to combine physical components into a wide range of cluster configurations, providing you with flexibility in building a cluster that meets your processing requirements. Figure 11 on page 41 shows an example of an HACMP cluster. Other HACMP clusters can look very different, depending on the number of processors, the choice of networking and disk technologies, and so on.
Nodes form the core of an HACMP cluster. A node is a processor that runs both AIX and HACMP. HACMP supports pSeries uniprocessor and symmetric multiprocessor (SMP) systems, and the Scalable POWERParallel processor (SP) systems as cluster nodes. The HACMP, an SMP system looks just like a uniprocessor. SMP systems provide a cost-effective way to increase cluster throughput. Each node in the cluster can be a large SMP machine, extending an HACMP cluster beyond the limits of a single system and allowing thousands of clients to connect to a single database.

In an HACMP Cluster, up to 32 computers or nodes cooperate to provide a set of services or resources to other remote clients. Clustering these servers to back up critical applications is a cost-effective high availability option. A business can use more of its computing power, to ensure that its critical applications resume running after a short interruption caused by a hardware or software failure.

In an HACMP cluster, each node is identified by a unique name. A node might own a set of resources (disks, volume groups, filesystems, networks, network...
addresses, and applications). Typically, a node runs a server or a “back-end” application that accesses data on the shared external disks.

HACMP supports from 2 to 32 nodes in a cluster, depending on the disk technology used for the shared external disks. A node in an HACMP cluster has several layers of software components.

Shared external disk devices
Each node must have access to one or more shared external disk devices. A shared external disk device is a disk physically connected to multiple nodes. The shared disk stores mission-critical data, typically mirrored or RAID-configured for data redundancy. A node in an HACMP cluster must also have internal disks that store the operating system and application binaries, but these disks are not shared.

Depending on the type of disk used, HACMP supports two types of access to shared external disk devices: non-concurrent and concurrent access.

- In non-concurrent access environments, only one connection is active at any time, and the node with the active connection owns the disk. When a node fails, disk takeover occurs when the node that currently owns the disk leaves the cluster and a surviving node assumes ownership of the shared disk.
- In concurrent access environments, the shared disks are actively connected to more than one node simultaneously. Therefore, when a node fails, disk takeover is not required.

Networks
As an independent, layered component of AIX, HACMP is designed to work with any TCP/IP-based network. Nodes in an HACMP cluster use the network to allow clients to access the cluster nodes, enable cluster nodes to exchange heartbeat messages, and, in concurrent access environments, serialize access to data.

HACMP defines two types of communication networks, characterized by whether these networks use communication interfaces based on the TCP/IP subsystem (TCP/IP-based), or communication devices based on non-TCP/IP subsystems (device-based).

Clients
A client is a processor that can access the nodes in a cluster over a local area network. Clients each run a front-end or client application that queries the server application running on the cluster node.

HACMP provides a highly-available environment for critical data and applications on cluster nodes. Note that HACMP does not make the clients themselves highly available. AIX clients can use the Client Information (Clinfo) services to receive notification of cluster events. Clinfo provides an API that displays cluster status information. The /usr/es/sbin/cluster/clstat utility, a Clinfo client provided with HACMP, provides information about all cluster service interfaces.

UNIX cluster overview
This section describes the procedure for granting high availability using High-Availability Cluster Multi-processing (HACMP), on the AIX, UNIX, and Linux for IBM operating systems. It is divided into the following subsections:

- “Prerequisite knowledge” on page 43
- “Standby and takeover configurations” on page 43
- “Design limitations” on page 46
Prerequisite knowledge

To understand the topics in this section, you must be familiar with Tivoli Workload Scheduler and HACMP clusters:

**Tivoli Workload Scheduler**
For an overview of Tivoli Workload Scheduler see the *Tivoli Workload Scheduler: Planning and Installation Guide*

**HACMP clusters**

Standby and takeover configurations

There are two basic types of cluster configuration:

**Standby**
This is the traditional redundant hardware configuration. One or more standby nodes are set aside idling, waiting for a primary server in the cluster to fail. This is also known as hot standby. From now on, we refer to an active/passive configuration to mean a two-node cluster with a hot standby configuration.

**Takeover**
In this configuration, all cluster nodes process part of the cluster's workload. No nodes are set aside as standby nodes. When a primary node fails, one of the other nodes assumes the workload of the failed node in addition to its existing primary workload. This is also known as mutual takeover.

Typically, implementations of both configurations will involve shared resources. Disks or mass storage such as a Storage Area Network (SAN) are most frequently configured as a shared resource.

As shown in [Figure 12 on page 44](#) Node A is the primary node, and Node B is the standby node currently idling. Although Node B has a connection to the shared mass storage resource, it is not active during normal operation.
After Node A fail over to Node B, the connection to the mass storage resource from Node B will be activated, and because Node A is unavailable, its connection to the mass storage resource is inactive. This is shown in Figure 13.

By contrast, in the following a takeover configuration, both Node A and Node B access the shared disk resource simultaneously. For Tivoli Workload Scheduler high-availability configurations, this usually means that the shared disk resource has separate, logical file system volumes, each accessed by a different node. This is illustrated in Figure 14 on page 45.
During normal operation of this two-node highly available cluster in a takeover configuration, the filesystem Node A FS is accessed by App 1 on Node A, and the filesystem Node B FS is accessed by App 2 on Node B. If either node fails, the other node takes on the workload of the failed node. For example, if Node A fails, App 1 is restarted on Node B, and Node B opens a connection to filesystem Node A FS. This is illustrated in Figure 15.

Takeover configurations are more efficient than standby configurations with hardware resources because there are no idle nodes. Performance can degrade after a node failure, however, because the overall load on the remaining nodes increases.
Design limitations

The following design limitations apply:

- "The master domain manager"
- "Tivoli Workload Scheduler commands"
- "Final status on running jobs"

The master domain manager

The Tivoli Workload Scheduler master domain manager is supported on the Cluster Virtual Server, but has two important limitations:

- The master domain manager runs the Final job stream which creates a new plan for the current production day. This process cannot be interrupted. An interruption might cause malfunctions and scheduling service interruptions. Only manual steps can be used to recover from such malfunctions or service interruptions. Because failover of the cluster group that contains the Tivoli Workload Scheduler resource stops the agent on the current node and starts it on a different node, if failover happens when the Final job stream runs, could be destructive.
- The Tivoli Workload Scheduler command-line utilities (conman, composer, and so on) are unaware of the cluster and if they are interrupted (through a failover of the cluster group that contains the Tivoli Workload Scheduler resource) they might corrupt some vital Tivoli Workload Scheduler information.

Tivoli Workload Scheduler commands

Any Tivoli Workload Scheduler command that is running during a failover is not automatically taken offline (unlike the main processes netman, mailman, batchman, and jobman) by the Tivoli Workload Scheduler cluster resource.

This is particularly problematical if the failover happens during an ad-hoc submission. The job submitted might remain in the ADDING state forever.

When Browse job log command via conman command line is active, the manual failover command is not working correctly and you must close all windows when the command is up and running.

Final status on running jobs

If a job is running during failover, its final state is ABEND with return code zero. Because the Jobman process is unable to retrieve the true final state of the job.

Supported configurations

This section describes the HACMP architecture we set up for the test environment followed by in-depth scenario descriptions.

For the scenarios, we have defined:

- 2 nodes
- 3 shared disks
- 1 volume group
- 1 application server
- 1 service IP address
- 1 resource group

We also configured the Heartbeat on Disk.
The Application Server contains the definition of the `start_tws.sh` and `stop_tws.sh` scripts that are described in detail in each section and that must be created on both nodes.

The `start_tws.sh` and `stop_tws.sh` scripts are located in `TWA_home/TWS/config` and you must customize them by setting the `DB2_INST_USER` parameter. After customizing the scripts, move them to another directory because any later release or fix pack overwrites them.

The Ethernet configuration we implemented is the IP Replacement, to have 1 boot address for each node and the Service IP address replaces the active one. In this configuration, the boot address of the active node can no longer be reached so, to avoid problems during the Tivoli Workload Scheduler installation, we configured an alias on the Ethernet network interface with the value of the boot address itself. Using the IP Aliasing configuration this additional step is unnecessary.

The following HACMP scenarios are supported with Tivoli Workload Scheduler:

- “Scenario: Shared disk, passive–active failover on a master domain manager”
- “Shared Disk, Passive – Active Failovers on Fault-Tolerant Agent” on page 49
- “Switching Domain Managers” on page 49

As an additional scenario we can also consider the possibility to have on the Master Domain Manager a local or a remote DB2® instance.

**Scenario: Shared disk, passive–active failover on a master domain manager**

This scenario describes how to configure Tivoli Workload Scheduler and a remote or local DB2 database so that a HACMP cluster is able to manage the failover of the active master domain manager.

**Configuring Tivoli Workload Scheduler and a remote DB2 database**

The following procedure explains how to configure Tivoli Workload Scheduler and a remote DB2 database so that a passive, idle node in the cluster can take over from an active master domain manager that has failed. The prerequisite for this procedure is that you have already configured HACMP.

Ensure that Installation Manager is installed in folders mounted on a shared disk and shared among all the cluster nodes. If you do not change them during the installation, the default installation directories are:

```
/var/ibm/InstallationManager
/opt/IBM/InstallationManager
/opt/IBM/IMShared
```

Install Tivoli Workload Scheduler using one of the installation methods described in *Tivoli Workload Scheduler: Planning and Installation Guide*.

Install Tivoli Workload Scheduler using a shared Installation Manager instance, so that any patching, installing, uninstalling, or upgrading activity can be performed on any Tivoli Workload Scheduler node.

During the installation, perform the followings configuration steps:
1. Create the same TWS administrator user and group on all the nodes of the cluster. Ensure that the user has the same ID on all the nodes and points to the same home directory on the shared disk where you are going to install Tivoli Workload Scheduler.

**Example:** You want to create the group named `twsadm` for all Tivoli Workload Scheduler administrators and the TWS Administrator user named `twsusr` with user ID 518 and home `/cluster/home/twsusr` on the shared disk:

```bash
mkgroup id=518 twsadm
mkuser id=518 pgrp=twsadm home=/cluster/home/twsusr twsusr
passwd twsusr
```

To install Tivoli Workload Scheduler in a directory other than the user home on the shared disk, ensure that the directory structure is the same on all nodes and that the `useropts` file is available to all nodes. Ensure also that the user has the same ID on all the nodes of the cluster.

2. Start the node that you want to use to run the installation of Tivoli Workload Scheduler and set the parameters so that HACMP mounts the shared disk automatically.

3. Install the DB2 administrative client on both nodes or on a shared disk configuring it for failover as described in DB2 documentation.

4. Create the `db2inst1` instance on the active node to create a direct link between Tivoli Workload Scheduler and the remote DB2 server.

5. Proceed with the Tivoli Workload Scheduler installation, using `twsuser` as the home directory and the local `db2inst1` instance.

After you installed Tivoli Workload Scheduler, run the cluster collector tool to automatically collect files from the active master domain manager. These files include the registry files, the Software Distribution catalog, and the Tivoli Workload Scheduler external libraries. The cluster collector tool creates a `.tar` file containing the collected files. To copy these files on the passive nodes, you must extract this `.tar` file on them.

To configure Tivoli Workload Scheduler for HACMP, perform the following steps:

1. Run the cluster collector tool.

2. From `TWA_home/TWS/bin`, run `.!/twsClusterCollector.sh -collect -tarFileName tarFileName`

   where `tarFileName` is the complete path where the archive is stored.

3. Copy `tws_user_home/useropts_twsuser` from the active node to the passive master domain manager from both the root and user home directories, to the other nodes.

4. Replace the node hostname with the service IP address for the master domain manager definitions, the WebSphere Application Server, the Dynamic workload broker and the agent, as described in the *Administration Guide*, section Changing the workstation host name or IP address.

5. Copy the `start_tws.sh` and `stop_tws.sh` scripts from `TWA_home/TWS/config` to the `TWA_home` directory.

6. Customize the `start_tws.sh` and `stop_tws.sh` scripts by setting the `DB2_INST_USER` variable.

7. Try the `start_tws.sh` and `stop_tws.sh` scripts to verify Tivoli Workload Scheduler starts and stops correctly.

8. Move the shared volume on the second cluster node (if you have already defined the cluster group, you can move it by using the `clRGmove HACMP` command).
9. Run the collector tool to extract Tivoli Workload Scheduler libraries. From the 
   `{TWA_home}/TWS/bin` directory, run:
   
   ```
   ./twsClusterCollector.sh -deploy -tarFileName tarFileName
   ```

   Where `tarFileName` is the complete path where the archive is stored.

10. Configure a new Application Controller resource on HACMP using the 
    customized `start_tws.sh` and `stop_tws.sh` scripts.

    When invoked by the HACMP during the failover, the scripts automatically start 
    or stop the WebSphere Application Server and Tivoli Workload Scheduler, and 
    link or unlink all the workstations.

**Local DB2**

This scenario is based on all the steps described in “Configuring Tivoli Workload 
Scheduler and a remote DB2 database” on page 47, but you must perform some 
additional steps.

1. Install the DB2 locally on both the nodes or on the shared disk, without 
   creating a new instance.

2. Create a new instance on the shared disk, define all the DB2 users also on the 
   second node, and modify the following two files:
   - `/etc/hosts.equiv`
     Add a new line with just the Service IP address value.
   - `<db2-instance-home>/sqllib/db2nodes.cfg`
     Add a new line similar to the following line:
     ```
     0 <Service IP address> 0
     ```

In this scenario, customize the `start_tws.sh` and `stop_tws.sh` scripts by setting the 
`DB2_INST_USER` parameter used to run the start and stop of the DB2 instance 
during the “failover” phase.

The only difference concerns the monman process used for the Event Driven 
Workload Automation feature, so that there is an additional step either in the 
`start_tws.sh` or `stop_tws.sh` script for starting or stopping that service. In Tivoli 
Workload Scheduler there are the additional steps regarding the stop and start of 
the dynamic agent.

**Shared Disk, Passive – Active Failovers on Fault-Tolerant Agent**

This scenario is almost the same as “Scenario: Shared disk, passive-active failover 
on a master domain manager” on page 47, but there are no additional steps to 
perform on the DB2 and WebSphere Application Server and the `start_tws.sh` and 
`stop_tws.sh` scripts run just the `link/unlink` and start or stop commands.

**Switching Domain Managers**

In this scenario, the DB2 database is installed on a remote server and the DB2 
administration client is installed on both nodes. The configuration is based on a 
Master installation on the first node and a Backup Master on the second one. Both 
nodes are connected to the same DB2 remote server.

No additional post-installation steps are required. You can left the `stop_tws.sh` 
script empty and create `start_tws.sh` from `{TWA_home}/TWS/config/switch_tws.sh`. In 
`switch_tws.sh`, you must set the `DB2_INST_USER`, `DB2_INST_PASSWD`, and 
`TWS_DB` variables.
The `start_tws.sh` script runs the switch manager command and, as an additional step, changes the workstation definition in DB2 in order to support more conveniently a switch that lasts longer than a production day.

### Upgrading from previous versions of the product

Instructions to follow before upgrading to V8.6:

- Read the system requirements.
- Read the *Tivoli Workload Scheduler: Planning and Installation Guide*, chapter Upgrading.
- If you are upgrading a master domain manager or a backup master domain manager, perform a backup of the database.
- If you are upgrading a critical production node, perform a backup of the entire installation directory.
- Ensure that all the product processes have been shut down.
- Consider the following information that applies to the master domain manager, backup master domain manager, the domain manager, the backup domain manager, the dynamic agents, and the fault-tolerant agents:

#### Using new features:

- The default authentication mechanism configured for the WebSphere Application Server included in the product is based on the Federated User Registry which supports the simultaneous use of more than one user registry. In earlier releases, the stand-alone user registry was used that can be local operating system, PAM, or LDAP. During a direct upgrade of a master domain manager or a backup master domain manager, the installation wizard attempts to reconfigure your authentication mechanism to use the Federated User Registry. If the reconfiguration fails, but all the other upgrade steps complete successfully, the upgraded master domain manager is configured to use a stand-alone user registry. This action is a temporary measure that allows you to access the master domain manager. Follow the instructions in the *Tivoli Workload Scheduler: Administration Guide*, section Configuring authentication to configure your authentication mechanism to use the Federated User Registry.

- When you upgrade from a version earlier than V8.5.1, the new dynamic agent is installed. This agent, if enabled, listen for incoming requests from the master domain manager on a TCP/IP port different from the one used by netman. Ensure that this port is accessible and reachable after you upgrade. The new version of the product requires the Common Inventory Technology (CIT). This component is installed automatically during the upgrade. If you are configuring an hot standby HACMP cluster ensure that it is installed also on the passive node. To install CIT, use the command shown in *Files and components installed on the local disk*.

Note that, the dynamic agent runs jobs based on resource availability. In a cluster, for example, you can configure the workload to run on the node that is holding the resource at that time. In this case, to run the job where the service is located, you no longer need to configure the agent to fail over (to follow the service). You can design your
workload according to a resource requirement, ensuring the logical resource that the jobs depend on is associated to the correct node every time the service fails over.

Files and components installed on the local disk

- The product binary files depend on some files that are installed on the local file system in the following directories:
  - /etc/TWS
  - /etc/TWA
  - .swdis

  This directory is the default Software Distribution directory. The product changes the directory specified in the **product_dir** property in the /etc/Tivoli/swdis.ini file.
  - /usr/Tivoli/TWS

  These files must be replicated on the passive node. The product provides the **twsClusterCollector.sh** utility to create a tar file with all external dependencies. Create a tar file and extract it on the passive node by using the **twsClusterCollector.sh** utility.

- The product embeds the new Common Inventory Technology (CIT). You must install CIT if you want to use the dynamic agent. You can install CIT manually on the passive node by using the following commands assuming that you downloaded the product images into the TWS_B6 directory:

  ```
  wdinstsp -D CIT_ExploiterID=TWA /TWS_B6/AIX/CIT_Preinstall.spb
  wdinstsp /TWS_B6/AIX/CIT.spb
  ```

After upgrading

Instructions to follow after upgrading to V8.6.

After you upgraded, perform the following actions:

- New features added to the product are enabled giving specific permissions to the users through the security file. During an upgrade, the product does not change your existing (customized) security file. You must modify the security file to include new security statements. See the *Tivoli Workload Scheduler: Planning and Installation Guide*, section [Upgrading a master domain manager or backup master domain manager instance](#).

  ```
  To configure the product to work with the HACMP service IP address, follow the instructions in the *Tivoli Workload Scheduler: Administration Guide*, section [Changing the workstation host name or IP address](#). By default the installation wizard uses the node host name.
  ```

- To start and stop the product, by using the **start_tws.sh** and **stop_tws.sh** scripts in the directory **TWA_home/TWS/config**. Customize these scripts to satisfy your environment requirements.
Appendix. Resolving desktop heap size problems on workstations with more than three agents

This appendix describes how to resolve the problem where the Windows desktop heap memory limitations cause processes to fail if there are more than three instances of Tivoli Workload Scheduler installed on a workstation in a Windows 2003 cluster environment.

Use this description whether you want to prevent the problem occurring (before installing the fourth agent instance) or if a problem has occurred caused by this limitation.

This section has the following topics

- "Problem description"
- "Solutions" on page 54
- "Implementing the solutions" on page 55

Problem description

The problem occurs because of the way Windows handles its desktop heap memory, and the way Tivoli Workload Scheduler creates desktops. In the security context, a desktop is used to encapsulate Windows processes, preventing the process from performing unauthorized activities.

The total amount of memory available for the creation of desktops is determined by a Windows registry entry called:

HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Session Manager \Memory Management\SessionViewSize

The default value is 20 MB.

The share of that buffer for each desktop is determined by a Windows registry entry called:

HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Session Manager \SubSystems\Windows

For example, the value of this entry might be:

%SystemRoot%\system32\csrss.exe ObjectDirectory=\Windows SharedSection= 1024,3072,512 Windows=On SubSystemType=Windows ServerDll=basesrv, 1 ServerDll=winsrv:UserServerDllInitialization,3 ServerDll=winsrv: ConServerDllInitialization,2 ProfileControl=Off MaxRequestThreads=16

In this entry, after the keyword SharedSection, there are three comma-separated memory entries (in KBs):

**Common memory (first entry)**

Defines the shared heap size common to all desktops (1024 in the example).

**Interactive desktop memory (second entry)**

Defines the extra desktop heap memory assigned to each interactive
process (3072 in the example). For example, the process which is in foreground at the moment. There are normally three interactive processes running at any one time.

Non-interactive desktop memory (third entry)
Defines the extra desktop memory assigned to non-interactive processes (512 in the example). For example, any process running in background.

Tivoli Workload Scheduler processes make the following use of desktops:

Tivoli Netman Windows service
Creates a non-interactive desktop shared between all agents running on the physical computer.

Tivoli Token Service Windows service
Creates a non-interactive desktop for the TWSUser for each agent.

Job manager (jobmon.exe)
Creates a non-interactive desktop for all jobs launched by each agent.

Thus, for each extra agent, three non-interactive desktops are created. The problem occurs when Windows uses up all the memory for creating desktops.

---

**Solutions**

To reduce the risk that a Tivoli Workload Scheduler process cannot find sufficient memory to create a desktop, do one, or more, of the following:

- “Modify the shared heap buffer sizes”
- “Configure the Tivoli Workload Scheduler Windows service to start as a local system account”
- “Customize the desktop name so that it is reused” on page 55

**Modify the shared heap buffer sizes**

If you reduce the size of the common or interactive memory, you leave more memory available for non-interactive desktops. However, reducing the sizes of either of these might cause performance problems. Microsoft sets these values by default because their tests show that these are the required values. You are not recommended to change these values.

Reducing the memory used for a non-interactive desktop will allow more desktops to be created. Individual processes that require more memory might be impacted, but most processes will run successfully. If your default non-interactive desktop memory (third entry) is 512, try reducing it to 256. See “Modify the Windows registry entries that determine the heap size” on page 56 for how to do it.

**Configure the Tivoli Workload Scheduler Windows service to start as a local system account**

By default, the Tivoli Workload Scheduler Windows service is configured for the TWSUser of each agent. By changing it to start as a local system account, only one desktop instance is created on the computer, not one per agent. The solution is implemented as follows:

- For agents installed with version 9.1, or later, this change is achieved by using the optional installation parameter –sharedDesktop
• For agents being installed at earlier versions, or agents already installed, make the change manually. See “Modify configuration of Windows service” for how to do it.

Customize the desktop name so that it is reused

When Jobmon opens a desktop, it allocates a unique name to the desktop, ensuring that a different desktop is created for each agent. However, if it creates the desktop using the name of a desktop already open, that process will open inside the existing desktop. To avoid this, you need to customize the name that Jobmon uses when it creates its desktop. By using the same name for all agents, each instance of Jobmon opens in the same desktop.

To ensure that this option is effective, the supplied name must be the same for at least two of the agents installed. The more agents that are run using the same shared desktop, the more memory will be available for desktop creation. However, if too many agents use the same shared desktop, there might be an impact on the ability of Windows to manage the jobs running in the shared desktop correctly. In this case, you might want to make a compromise. For example, if you had four agents installed on the same computer, you could choose to have pairs of agents share the same desktop.

The solution is implemented as follows:

• For agents installed with version 9.1, or later, this change is achieved by using the optional installation parameter –sharedDesktop. If you add this option without an argument, the installation applies the default name of TWS_JOBS_WINSTA. Otherwise supply your own name, for example, –sharedDesktop name="my windows desktop name".

See “twsClusterAdm command with examples of usage” on page 18 for how to do it.

• For agents being installed at earlier versions, or agents already installed, make the change manually. See “Modify localopts to supply a shared desktop name” on page 56 for how to do it.

Implementing the solutions

There are several possible solutions. Choose the one that is best for your circumstances:

• “Modify configuration of Windows service”
• “Modify the Windows registry entries that determine the heap size” on page 56
• “Modify localopts to supply a shared desktop name” on page 56

Modify configuration of Windows service

To modify the Tivoli Workload Scheduler Windows service to open as a local account, do the following:

1. From the Start button, select the Services panel (for example, select Programs → Administrative Tools → Services)
2. Select the Tivoli Workload Scheduler service and double-click it to edit it
3. Select the Log on tab
4. Click Local System account and then Apply
5. Right-click the service and select Stop
6. When the service has stopped, right-click it again and select Start
7. Check that the service has started correctly and close the Services window.
Modify the Windows registry entries that determine the heap size

To modify the Windows registry entries that determine the heap size, run `regedit.exe` and modify the key:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Session Manager \SubSystems\Windows
```

The default data for this registry value will look something like the following (all on one line):

```
%SystemRoot%\system32\csrss.exe ObjectDirectory=\Windows SharedSection=1024,3072,512 Windows=On SubSystemType=Windows ServerDll=basesrv, 1 ServerDll=winsrv:UserServerDllInitialization,3 ServerDll=winsrv: ConServerDllInitialization,2 ProfileControl=Off MaxRequestThreads=16
```

The numeric values following `SharedSection=` control how the desktop heap is allocated. These `SharedSection` values are specified in kilobytes. See [“Problem description” on page 53](#) for a description of the values.

The third `SharedSection` value (512 in the above example) is the size of the desktop heap for each non-interactive desktop. Decrease the value to 256 kilobyte.

**Note:** Decreasing any of the `SharedSection` values will increase the number of desktops that can be created in the corresponding window stations. Smaller values will limit the number of hooks, menus, strings, and windows that can be created within a desktop. On the other hand, increasing the `SharedSection` values will decrease the number of desktops that can be created, but will increase the number of hooks, menus, strings, and windows that can be created within a desktop. This change will only take effect after you reboot the cluster nodes.

**Modify localopts to supply a shared desktop name**

To use a shared desktop name for an agent already installed, do the following:

1. Open the `localopts` file for the agent in question (see the Tivoli Workload Scheduler: Planning and Installation Guide for the location of this file)
2. Add the key `jm windows station name = <my_name>`. Ensure that `<my_name>` is the same name as used in another agent to save desktop memory.
3. Save the file.
4. Stop and restart Tivoli Workload Scheduler to make the change effective.
Notices

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

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not 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.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, 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 websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.
IBM may use or distribute any of the information you 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 document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

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](http://www.ibm.com/legal/copytrade.shtml).

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

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

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

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

Special characters

–ask, twsClusterAdm.exe argument 23
–changeResName, twsClusterAdm.exe parameter 23
–dll, twsClusterAdm.exe parameter 22
–force, twsClusterAdm.exe argument 23
–force, twsClusterAdm.exe parameter 22
–new, twsClusterAdm.exe parameter 20
–notwsinst, twsClusterAdm.exe parameter 22
–sharedDesktop, twsClusterAdm.exe parameter 22
–twsupd, twsClusterAdm.exe argument 23
–uninst, twsClusterAdm.exe parameter 22
–update, twsClusterAdm.exe parameter 23

A
abend state 36
accessibility x
agent, installing in a cluster-aware configuration 12
arguments, to twsClusterAdm.exe 20
ask, twsClusterAdm.exe argument 23

B
backup domain manager
Tivoli Workload Scheduler 1

C
changeResName, twsClusterAdm.exe parameter 23
check_interval, twsClusterAdm.exe argument 21
clu_offline.log 35
clu_online.log 35
cluster Administrator console, using 32
Cluster Administrator extension
installing 34
overview 33
Parameters tab 34
Cluster HACMP upgrading
to version 8.6 50, 51
cluster instance name
local option 31
modifying
in the localopts file 32
in the ShutDown_clu.cmd script 33
in the StartUp_clu.cmd script 33
cluster resource dependencies (continued)
  Physical Disk 29
  commands
    cluster res 31
    ShutDown_clu.cmd 31
    Shutdown.cmd 13
    StartUp_clu.cmd 31
tws_env.cmd 13
twsClusterAdm.exe 18
twsClusterAdmin.exe, local option 31
compatibility
  with previous versions 6
  configuration 43
Conman shut 31
Conman start 31
contents, Windows 2003 Cluster Enabler 9
disk, twsClusterAdm.exe argument 20
core dump 37
custom resource DLL 8
entry-points
  IsAlive 8
  Offline 8
  Online 8
  Terminate 8

D
disk, twsClusterAdm.exe argument 20
Dlls
  custom resource DLL 8
DLL files
  ITWSResources.dll 9
  ITWSWorkstationEx.dll 9
dll, twsClusterAdm.exe parameter 22
domain, twsClusterAdm.exe argument 20, 22
Dynamic Workload Console accessibility x

E
education x
error 1314 36
examples
twsClusterAdm.exe 24

F
failover, twsClusterAdm.exe argument 21
fault-tolerance
definition 1
force, twsClusterAdm.exe argument 23
force, twsClusterAdm.exe parameter 22

G
glossary ix
group, twsClusterAdm.exe argument 20

H
HACMP cluster upgrading
  to version 8.6 50, 51
high availability
types of 1
high-availability definition 1
hosts, twsClusterAdm.exe argument 20, 23

I
installation
  installing in a cluster 9
  prerequisites 11
installing
  Cluster Administrator extension 34
  Tivoli Workload Scheduler agent 12
  Windows Cluster Enabler 9
installing in a cluster 9
IP Address 29
ip, twsClusterAdm.exe argument 20
Isalive, twsClusterAdm.exe argument 21
ITWSExtInst.cmd 9
ITWSResources.dll 9
ITWSWorkstationEx.dll 9

L
localopts file
  modifying 32
lookalive, twsClusterAdm.exe argument 21

M
master domain manager 46
  not supported on Cluster Virtual Server
  reasons why 6

N
name, twsClusterAdm.exe argument 22
net, twsClusterAdm.exe argument 20
Network Name 29
new_resource_instance_name,
twsClusterAdm.exe argument 24
new, twsClusterAdm.exe parameter 20
notwsinst, twsClusterAdm.exe parameter 22
Tivoli Workload Scheduler: High Availability Cluster Environments

O
operating systems 6
opts, twsClusterAdm.exe argument 21

P
Parameters tab
Cluster Administrator extension 34
parameters, to twsClusterAdm.exe parameter 20
path, twsClusterAdm.exe argument 22
physical Disk 29
prerequisites 11
publications ix
pwd, twsClusterAdm.exe argument 20

R
recovery action in case of failover
allowing for 1
Replace a process level token
security policy 12, 36
res, twsClusterAdm.exe parameter 20
rename, twsClusterAdm.exe argument 21
resource instance name
changing 32
resource_instance_name,
twsClusterAdm.exe argument 23
resource, twsClusterAdm.exe argument 23
roll back or recovery action
allowing for 1

S
scripts
ShutDown_clu.cmd 31
StartUp_clu.cmd 31
security policy
Replace a process level token 12, 36
sharedDesktop, twsClusterAdm.exe parameter 22
ShutDown_clu.cmd 31
ShutDown.cmd 31, 31
ShutDownLwa.cmd 31
StartUp_clu.cmd 31
StartUp.cmd 31
StartUpLwa.cmd 31
syntax
twsClusterAdm.exe 19

T
taxover 43
tcpport, twsClusterAdm.exe argument 21
technical training 7
Tivoli Workload Scheduler 42, 43, 46
authorization 7
backup domain manager 1
benefits 39
clients 42
Cluster Administrator extension 33
installing 34
Tivoli Workload Scheduler (continued)
Cluster Administrator extension (continued)
overview 33
cluster environment
integrating into 5
cluster resource dependencies 29
command-line
not automatically taken offline
during a failover 6, 46
compatibility
with previous versions 6
configuring
with twsClusterAdm 18
HACMP 39, 40, 41
in the Windows cluster
environments 29
integrating into
cluster environment 5
make cluster-aware 13
master domain manager 46
not supported on Cluster Virtual
Server 6
networks 42
nodes 41
physical components 40
product design limitations 5, 46
recovery action in case of failover 1
resource instance name
changing 32
roll back or recovery action 1
security authentication 7
shared external disk 42
standby 43
starting 31
cluster res command 31
stopping 31
cluster res command 31
troubleshooting 35
abend state 36
core dump 37
error 1314 36
trace files 35
uninstalling 34
Unix 39
where to find information 5, 43
Windows 5
Windows 2003 Cluster Enabler 7
components 7
prerequisites 6
Tivoli Workload Scheduler agent
installing
in a cluster-aware
configuration 12
upgrading manually
in a cluster-aware
configuration 17
Tivoli Workload Scheduler agent from V8.5.1 and fix packs
upgrading
in a cluster-aware configuration
with twsClusterUpg 15
Tivoli Workload Scheduler agent from V8.6 and fix packs
upgrading
in a cluster-aware configuration
with twsClusterUpg 15
trace files 35
clu_offline.log 35
clu_online.log 35
cluster.log 36
training
technical x
troubleshooting 35
abend state 36
error 1314 36
jobmon
core dump 37
trace files 35
tws_env.cmd 13
twsClusterAdm
Tivoli Workload Scheduler agent 18
twsClusterAdm.exe 9, 13, 18
arguments 20
examples 24
parameters 20
syntax 19
twsClusterAdmin.exe
cluster instance name
local option 31
twsClusterUpg syntax 15
twshome, twsClusterAdm.exe argument 20
twsupd, twsClusterAdm.exe argument 23

U
uninst, twsClusterAdm.exe parameter 22
Unix 39
UNIX
where to find information 43
update, twsClusterAdm.exe parameter 23
upgrading
Tivoli Workload Scheduler agent 17
upgrading in HACMP cluster
to version 8.6 50, 51
upgrading manually 17
upgrading with twsClusterUpg
Tivoli Workload Scheduler agent from V8.5.1 and fix packs 15
Tivoli Workload Scheduler agent from V8.5.1 or V8.6 and their fix
packs 15
user, twsClusterAdm.exe argument 20, 23
utility
Shutdown.cmd 13

V
version 8.6
cluster HACMP upgrading to 50, 51

W
Windows 5
Windows 2003 and 2008 Cluster Enabler contents
ITWSExInst.cmd 9
ITWSResources.dll 9
Windows 2003 and 2008 Cluster Enabler
(continued)
    contents (continued)
    ITWSWorkstationEx.dll  9
twsClusterAdm.exe  9
Windows 2003 Cluster Enabler
    components  7
    contents  9
Windows 2003 clusters
    where to find information  5
Windows Cluster Enabler
    components
        custom resource DLL  8
    installing  9
    installing in a cluster  9
    prerequisites  11