

HCL Workload Automation
High Availability Cluster Environments
Version 10.2.5



Note

Before using this information and the product it supports, read the information in [Notices on page lx](#).

This edition applies to version 10, release 2, modification level 5 of HCL Workload Automation (program number 5698-T09) and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

List of Figures.....	v	Appendix A. Resolving desktop heap size problems on workstations with more than three agents.....	55
List of Tables.....	vi	Problem description.....	55
About this publication.....	vii	Solutions.....	56
What is new in this release.....	vii	Modify the shared heap buffer sizes.....	56
Accessibility	vii	Configure HCL Workload Automation Windows service to start as local system account.....	57
Chapter 1. Types of high availability.....	8	Customize the desktop name so that it is reused.....	57
Agent availability versus job availability.....	8	Implementing the solutions.....	57
Backup domain manager.....	8	Modify configuration of Windows service.....	58
Chapter 2. HCL Workload Automation with Windows cluster	10	Modify the Windows registry entries that determine the heap size.....	58
Windows® cluster overview.....	10	Modify localopts to supply a shared desktop name.....	58
HCL Workload Automation with Windows cluster environments.....	10	Notices.....	lx
Prerequisite knowledge.....	10	Index.....	64
Design limitations.....	11		
Supported operating systems.....	11		
Security and Authentication.....	12		
Windows Cluster Enabler.....	12		
Components.....	12		
Installation and configuration.....	14		
Upgrading agents to a new version and available fix packs.....	19		
twClusterAdm command.....	21		
Operating HCL Workload Automation in Windows cluster environments.....	35		
Cluster Administrator extension.....	38		
Uninstalling HCL Workload Automation.....	39		
Troubleshooting.....	40		
Traces.....	40		
Error 1314 taking online the resource and Workstation does not link.....	41		
Resource instance reports fail status or user jobs go in abend state.....	42		
Windows Report panel with Jobmon.exe core dump.....	42		
Cluster: IP validation error on Netman stdlist.....	42		
Chapter 3. HCL Workload Automation with HACMP.....	43		
High-Availability Cluster Multi-Processing.....	43		
Benefits.....	43		
Physical components of an HACMP cluster.....	44		
UNIX cluster overview.....	46		
Prerequisite knowledge.....	46		
Standby and takeover configurations.....	46		
Design limitations.....	48		
Supported configurations.....	49		
Upgrading from previous versions of the product.....	52		

List of Figures

Figure 1: Main components of the HCL Workload Automation Cluster Enabler.....	13
Figure 2: Installing in a cluster.....	15
Figure 3: Clusters in an HCL Workload Automation network.....	15
Figure 4: File Server example on Windows Server 2008.....	19
Figure 5: Resource Dependencies tab (Windows Server 2008).....	36
Figure 6: New Properties Parameters tab.....	39
Figure 7: Shared disk with mirror.....	44
Figure 8: Active-Passive configuration in normal operation.....	47
Figure 9: Failover on Active-Passive configuration.....	47
Figure 10: Logical file system volumes.....	48
Figure 11: Failover scenario.....	48

List of Tables

Table 1: Monitoring options in your cluster environment..... 25

About this publication

This publication describes how Windows™, UNIX®, Linux®, and HACMP for AIX clusters fit into the topology of HCL Workload Automation. This publication also describes the enhancements to HCL Workload Automation to support the clustering and high-availability environment based on Microsoft™ Windows™.

What is new in this release

Learn what is new in this release.

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

New or changed content is marked with revision bars.

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 detailed information, see the appendix about accessibility in the *HCL Workload Automation User's Guide and Reference*.

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 on page 8](#)

Agent availability versus job availability

Having HCL Workload Automation 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 HCL Workload Automation 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 HCL Workload Automation 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, HCL Workload Automation 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® Workload Scheduler and IBM® Tivoli® Framework*).

HACMP for AIX scenario - Backup domain manager

HCL Workload Automation 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 HCL Workload Automation network
- A high-availability product addresses many of the coding issues that surround detecting hardware failures
- Implementing high-availability for fault-tolerant agents cannot be accomplished using the backup domain manager feature

Chapter 2. HCL Workload Automation with Windows cluster

This section contains information on the following topics:

- [Windows cluster overview on page 10](#)
- [Enabling HCL Workload Automation to work in a Windows Cluster environment on page 12](#)
- [Troubleshooting on page 40](#)

Windows® cluster overview

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

- [HCL Workload Automation with Microsoft Windows cluster environments on page 10](#)
- [Prerequisite knowledge on page 10](#)
- [Design limitations on page 11](#)
- [Supported operating systems on page 11](#)
- Compatibility, upgrade, and coexistence
- [Security and Authentication on page 12](#)

HCL Workload Automation with Microsoft® Windows cluster environments

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

An HCL Workload Automation agent configured to work in a Windows® cluster environment can be connected to both the distributed and the end-to-end network configurations.

Prerequisite knowledge

To understand this document, you must be knowledgeable about HCL Workload Automation and Microsoft® Windows clusters:

HCL Workload Automation

For an overview of HCL Workload Automation, see *Planning and Installation Guide*.

Microsoft® Windows clusters

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

Design limitations

The following design limitations apply:

- [The master domain manager on page 11](#)
- [HCL Workload Automation commands on page 11](#)
- [Use with multiple agents on page 11](#)

The master domain manager

The HCL Workload Automation 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 HCL Workload Automation 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 HCL Workload Automation 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 HCL Workload Automation resource) they might corrupt some vital information for HCL Workload Automation.

HCL Workload Automation commands

Any HCL Workload Automation command that is running during a failover is not automatically taken offline (unlike the main processes `netman`, `mailman`, `batchman`, and `jobman`) by the HCL Workload Automation 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 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 55](#).

Supported operating systems

The HCL Workload Automation Windows Cluster Enabler is available for 64-bit Windows® systems.

To obtain the latest information about supported Windows versions, see [Prerequisites on page 16](#).

Security and Authentication

The usual HCL Workload Automation security authentication and authorization mechanism applies.

Enabling HCL Workload Automation to work in a Windows Cluster environment

About this task

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

- [HCL Workload Automation Windows Cluster Enabler components on page 12](#)
- [Installation and configuration on page 14](#)
- [Upgrading cluster nodes manually on page 20](#)
- [twClusterAdm command with examples of usage on page 21](#)
- [Operating HCL Workload Automation in Windows cluster environment on page 35](#)
- [Uninstalling HCL Workload Automation on page 39](#)

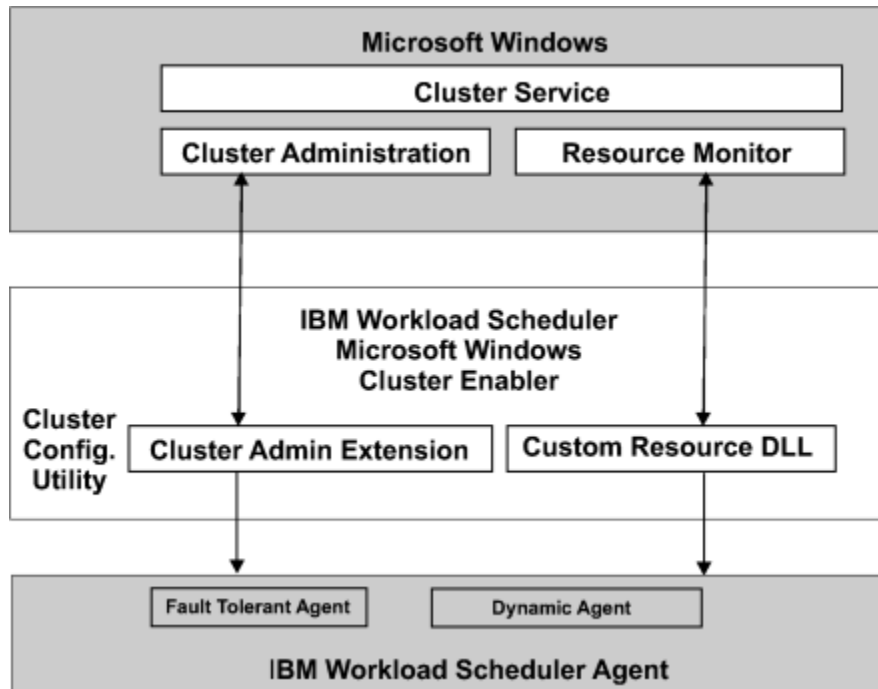
HCL Workload Automation Windows Cluster Enabler components

About this task

The HCL Workload Automation Windows Cluster Enabler consists of the following elements:

- A utility to:
 - Install and remotely configure HCL Workload Automation on all the other nodes of the cluster
 - Install and configure the HCL Workload Automation cluster resource type for a given virtual server
- An HCL Workload Automation Manager Custom Resource type to manage cluster events for HCL Workload Automation instances (new DLLs)
- An HCL Workload Automation extension DLL to extend the Windows® Cluster Administration program

Figure 1. Main components of the HCL Workload Automation Cluster Enabler



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 shown in [Figure 1: Main components of the HCL Workload Automation Cluster Enabler on page 13](#), 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 HCL Workload Automation 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 HCL Workload Automation resources.

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

IsAlive

Determines if the HCL Workload Automation agent is currently active.

Offline

Performs a graceful shutdown of the HCL Workload Automation agent.

Online

Starts the HCL Workload Automation agent, links the agent to the network, and makes the resource available to the cluster.

Terminate

Performs an immediate shutdown of the resource.

The HCL Workload Automation Manager custom resource DLL is a bridge between the resource monitor (part of the Windows® cluster service) and the HCL Workload Automation 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

About this task

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

- [Windows Cluster Enabler on page 14](#)
- [Installing in a cluster on page 15](#)
- [Prerequisites on page 16](#)
- [Install and configure a new HCL Workload Automation agent on page 18](#)

The HCL Workload Automation Windows Cluster Enabler is installed automatically when you install HCL Workload Automation. A new folder, named `cluster`, is created within the HCL Workload Automation installation directory.

Windows Cluster Enabler

About this task

To enable the HCL Workload Automation to work in a windows cluster environment, the installation process provides the following files:

ITWSWorkstationEx.dll

The HCL Workload Automation Cluster Administrator extension. It adds a new property sheet and wizard pages for the HCL Workload Automation resource type to the Cluster Administrator console. See [HCL Workload Automation Cluster Administrator extension on page 38](#) for more details.

twscClusterAdm.exe

Used to install and configure HCL Workload Automation.

ITWSResources.dll

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

ITWSExInst.cmd

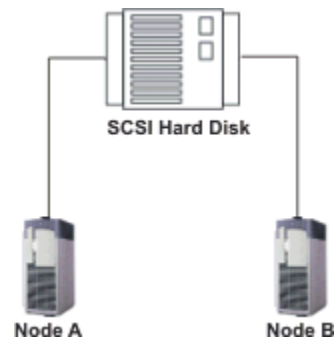
The sample script that registers the HCL Workload Automation Cluster Administrator extension.

Installing in a cluster

About this task

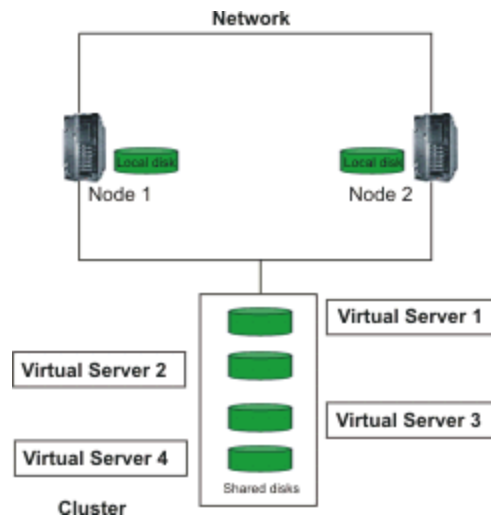
A minimal cluster configuration is composed of two nodes. Complying with the used disk technology, a Windows cluster can have from 2 to 36 nodes.

Figure 2. Installing in a cluster



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**.

Figure 3. Clusters in an HCL Workload Automation network



To have HCL Workload Automation working in a cluster environment you can:

- Install the HCL Workload Automation 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 HCL Workload Automation agent on one or more virtual servers if you need to schedule jobs on that virtual server. Cluster awareness is required.

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

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.

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

Prerequisites

About this task

The following are prerequisites for the correct setup of HCL Workload Automation on the cluster:

Windows Cluster Server

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

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 HCL Workload Automation ones.

To create the File Server Service, you can use the Failover Cluster Manager console. You must 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 configured Role for Windows 2012

A Role, formerly called File Server Service, contains at least the virtual IP address resource, the network name resource, and the physical disk resource. The Role can contain other application resources, not only HCL Workload Automation ones.

To create a File Server Role with type General Use, you can use the Failover Cluster Manager console. You must do the following:

- Create the File Server Role
- Associate the static IP address to the File Server Role

- Add the Shared Disk to the File Server Role created
- The system automatically adds to the File Server Role 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 HCL Workload Automation 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 HCL Workload Automation user. In this way the HCL Workload Automation custom resource that runs with the rights of the Cluster Administrator user, is able to stop, start, and link HCL Workload Automation. Restart the cluster nodes to have this change take effect. This operation is required only the first time you configure HCL Workload Automation to work in the Windows® cluster environments.

Install Microsoft® Visual C++ 2013 Redistributable Package (x86 or x64) and Microsoft® Visual C++ 2015-2019 Redistributable (x64 or x86) 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 above packages. The installation of the HCL Workload Automation 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:

1. Either download the above packages from Microsoft Download Center website or go to <http://www.microsoft.com> and search for the package by name. Download the following package files:
 - `vc_redist_x86_13.exe` VC++2013 Redistributable(x86) or `vc_redist_x64_13.exe` VC++2013 Redistributable(x64)
 - `vc_redist.x86_14.exe` VC++2015-2019 Redistributable(x86) or `vc_redist.x64_14.exe` VC++2015-2019 Redistributable(x64)
2. Copy the package to each node in the Cluster Virtual Server Group:
3. On each node in the group (other than the one where you install the cluster enabler), do the following:
 - a. Log on as Domain Administrator
 - b. Run the package executables.

Install and configure a new HCL Workload Automation agent

About this task

To install HCL Workload Automation in a cluster-aware configuration, use the following procedure:

1. Install the HCL Workload Automation 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 by using a user with Domain Administrator privileges.
 - c. Choose the Microsoft® Virtual Server (for Windows 2008) or Clustered Role (for Windows 2012) where you want to install the HCL Workload Automation agent.
 - d. Install HCL Workload Automation:
 - Specify a domain user as user for which you want to install HCL Workload Automation
 - Specify the shared disk that is associated to that Virtual Server (for Windows 2008) or Clustered Role (for Windows 2012) as destination directory
 - e. Install HCL Workload Automation.
 2. Make the HCL Workload Automation cluster aware:
 - a. Run the Windows® Command Prompt.
 - b. Move into the HCL Workload Automation home directory (on the shared disk).
 - c. Run `twc_env.cmd` to load the HCL Workload Automation environment variables.
 - d. Run `Shutdown.cmd` to stop HCL Workload Automation.
 - e. Move into the `cluster` directory.
 - f. Run the utility `twcClusterAdm.exe` to configure HCL Workload Automation remotely on all nodes of the cluster and to install the HCL Workload Automation Cluster Resource. See [Example 1: First installation of HCL Workload Automation in a Windows cluster environment on page 29](#) for an installation example.
-  **Note:** **ITWSWorkstation** is the name of the HCL Workload Automation cluster resource type. By default, when it is created by using the `twcClusterAdm` command line, the instance name is `ITWSWorkstation_<domain_name>_<user_name>`.
-  **Note:** During HCL Workload Automation 10.2.5 Cluster installation, the following parameters must not be specified in double-byte character set (DBCS) characters:
- User
 - Domain
3. Define a new workstation object on the master domain manager either by using **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 **ITWSWorkstation** resource instance that you created in [step 2f on page 18](#):

- a. Locate the **ITWSWorkstation** resource instance:

On Windows 2008 cluster:

From the **Failover Cluster Manager** console, find the File Server where the **ITWSWorkstation** resource instance is saved.

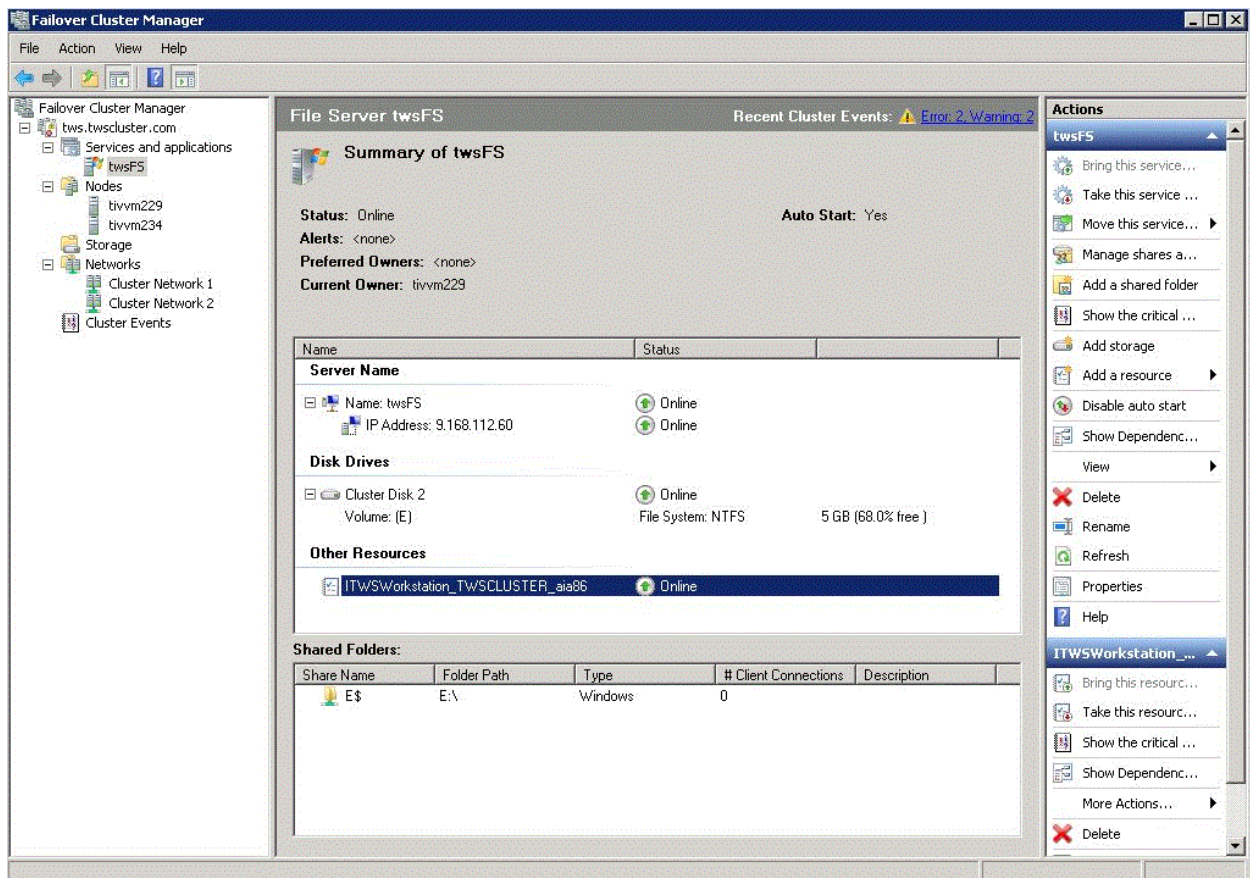
In the [Figure 4: File Server example on Windows Server 2008 on page 19](#) example, the `ITWSWorkstation_CLUSTER_aia86` resource instance is saved in the *twFS* File Server.

On Windows 2012 cluster:

From the **Failover Cluster Manager** console, find the Role where the **ITWSWorkstation** resource instance is saved.

- b. Right-click on the selected **ITWSWorkstation** resource instance and select **Bring Online**.

Figure 4. File Server example on Windows Server 2008



5. 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

You can upgrade cluster nodes to the latest version of the product either automatically, by running `twClusterUpg` script, or manually.

You can upgrade from:

- V9.1 or later

Upgrading cluster nodes manually

You can upgrade HCL Workload Automation agents in a cluster-aware configuration, from V9.1 or later, to the general availability (GA) version of the product, by running the following manual procedure.

About this task

1. Ensure your environment meets the prerequisites listed in [Prerequisites on page 16](#).
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 HCL Workload Automation 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:

```
cluster.exe node awsclupgrade.ditanode_name> /Pause
```

Where *awsclupgrade.ditanode_name*> is the name of the node to pause.

3. Set offline all the HCL Workload Automation resources belonging to the nodes of the cluster you paused to prevent HCL Workload Automation 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:

```
cluster.exe res awsclupgrade.ditaress_name> /Offline
```

Where *awsclupgrade.ditaress_name*> is the name of the resource to set offline.

4. Move to the directory where you downloaded the images of the general availability (GA) version.
5. 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:

```
cscript.exe twsinst.vbs -update -uname awsclupgrade.ditauser_name>
  -password awsclupgrade.ditaTWS_user_password>
  -acceptlicense yes
  -inst_dir TWS_install_dir -recovInstReg true
```

- b. Move to the *inst_dir* directory where you upgraded the cluster node and update the remote Windows Services and the resource DLL, by running the following command:

```
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 HCL Workload Automation user.



Note: If you added a node to your HCL Workload Automation cluster after installing HCL Workload Automation, 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 [5a on page 20](#) and [5b on page 20](#) for all the resources present on this node.

6. 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.

7. Bring online the HCL Workload Automation 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 HCL Workload Automation resource.

See [Example 10: Upgrading agents in a cluster-aware configuration on page 33](#) for an upgrade example.

twClusterAdm command with examples of usage

About this task

The **twClusterAdm** command configures your HCL Workload Automation agent type installations in the Microsoft® Windows cluster environment.

You can configure the following HCL Workload Automation agent type installations:

- Fault-tolerant agent only.
- Both fault-tolerant agent and dynamic agent.
- Dynamic agent only.

Both fault-tolerant agents and dynamic agents are monitored and can be involved in failover. For details, see the **ftaOff** and **lwaOn** monitoring options of **twClusterAdm** parameter **opts**, (described in [Syntax on page 22](#)).

If you want to install a domain manager on a cluster environment, you must specify the *link* option using the **twClusterAdm** parameter **opts** (described in [Syntax on page 22](#)). See [Example 9: First installation of domain manager in Windows cluster environment, specifying generic options on page 32](#) for a worked example.

The **twClusterAdm** command:

- Configures HCL Workload Automation on all the nodes of the cluster or on a new joining cluster node.
- Installs the new HCL Workload Automation Cluster resource type for a first time installation. The name of this new cluster resource type is `ITWSWorkstation`.
- Creates an instance of the HCL Workload Automation Cluster resource type within a cluster group.
- Removes HCL Workload Automation configuration from one or more nodes of the cluster.
- Upgrades the HCL Workload Automation 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 HCL Workload Automation resource DLL to the cluster nodes.
- Updates HCL Workload Automation services startup mode from automatic to manual.
- Installs HCL Workload Automation services and registry key on the other nodes (because the current services and registry key was installed by the normal product installation).
- Registers HCL Workload Automation Resource Types.
- Creates a new instance of the HCL Workload Automation resource within a given cluster group (virtual server) and updates `localopts` and `JobManager.ini` configuration files.

Syntax

About this task



Note: The order and the position of the **twClusterAdm** command parameters must be respected.

twClusterAdm.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>
[resname=<resource_instance_name>]
[check_interval=<TWS_polling_interval>]
[failover=yes|no]
[looksalive=<lookalive_interval>]
[isalive=<isalive_interval>]
[tcpport=<tcp_port>]
[opts=<generic_options>]
]
[-dll
[path=<DLL_resource_path>]
]
[-force]
[-sharedDesktop [name=<desktop_name>]]

```

twClusterAdm.exe -uninst

```

domain=<Windows_domain>

```

```

user=<TWS_user_name>
[hosts=<hostname1,hostname2...>]

```

twClusterAdm.exe -update

```

resource=<resource_instance_name>
[hosts=<hostname1,hostname2...>]
[ask={yes|no}]
[-force]
[-twsupd [pwd=<TWS_user_password>]]

```

twClusterAdm.exe -changeResName

```

"<resource_instance_name>"
"<new_resource_instance_name>"

```

Parameters and arguments

About this task

-new

The **-new** parameter configures HCL Workload Automation 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 HCL Workload Automation 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 HCL Workload Automation 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 HCL Workload Automation user. This parameter is mandatory if **-new** is specified.

hosts=<hostname1,hostname2...>

The host names of the cluster nodes where you want to configure HCL Workload Automation. 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 HCL Workload Automation is installed. This parameter is optional. If you do not specify this directory, the command discovers the installation directory.

-res

The **-res** parameter adds a new instance of the HCL Workload Automation resource type to an existing cluster group. It takes the following arguments:

group=<cluster_group_name>

The name of the group (Virtual Server) where HCL Workload Automation is configured as the cluster resource. This parameter is mandatory.

ip=<IP_resource_name>

The name of the cluster IP resource type that the HCL Workload Automation resource depends on. This parameter is mandatory.

net=<network_resource_name>

The name of the network resource type that the HCL Workload Automation resource depends on. This parameter is mandatory.

disk=<disk_resource_name>

The name of the disk resource type that the HCL Workload Automation 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: File Server example on Windows Server 2008 on page 19](#)). If this parameter is not supplied, the default value of `ITWSWorkstation_<domain_name>_<user_name>` is used.

failover=yes/no

If you specify **yes**, HCL Workload Automation can cause the failover of the virtual server group. If you do not specify this option HCL Workload Automation 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 HCL Workload Automation 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=<lookalive_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 **opts** parameter is used to specify a set of options. Each option is separated by a semicolon ";". The **opts** parameter accepts the following options:

- **Monitoring options:**

- **ftaOff**

Monitoring option: use this option to disable monitoring (and failover) of fault-tolerant agents.

- **lwaOn**

Monitoring option: use this option to enable monitoring (and failover) of dynamic agents.

For details about the monitoring options in your cluster environment, see [Table 1: Monitoring options in your cluster environment. on page 25](#).

Table 1. Monitoring options in your cluster environment.

Types of HCL Workload		
Automation agent installation in your cluster environment	Monitoring options for your agent installation	Result
▪ Agent installation composed of both fault-tolerant agent and dynamic agent	No options are specified	Only fault-tolerant agent is monitored (default).
	opts=lwaOn;ftaOff	Only dynamic agent is monitored.
▪ Agent installation composed of fault-tolerant agent only	opts=lwaOn	Both fault-tolerant agent and dynamic agent are monitored.
	opts=ftaOff	Neither fault-tolerant agent nor dynamic agent are monitored.
▪ Agent installation composed of dynamic agent only	No options are specified	Dynamic agent is monitored (default).
	opts=lwaOn;ftaOff	Dynamic agent is monitored.

Types of HCL Workload		
Automation agent installation in your cluster environment	Monitoring options for your agent installation	Result
	opts=lwaOn	Dynamic agent is monitored.
	opts=ftaOff	Dynamic 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=killjob;link=MASTERDM!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.

–dll

The **–dll** parameter specifies that the `ITWSResources.dll` that implements the new HCL Workload Automation resource type needs to be installed. This parameter is mandatory the first time you configure HCL Workload Automation 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 HCL Workload Automation 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 55](#) 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 HCL Workload Automation User. 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 HCL Workload Automation 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 HCL Workload Automation. Host names have to be separated by commas. This parameter is optional. If you do not specify this parameter, HCL Workload Automation is uninstalled from all nodes in the cluster except for the current node.

-update

The **-update** parameter updates the HCL Workload Automation resource DLL of an existing instance, and accepts the following arguments:

resource=<resource_instance_name>

The HCL Workload Automation 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 resource DLL upgrade can be performed automatically. Supply *yes* to ask that the resource upgrade is performed automatically, if the upgrade version is greater than the actual version or - **force** parameter is provided. Supply *no* to ask that the resource is not upgraded. This parameter is optional. If it is not specified, the installation stops and, in case the upgrade version is greater than the actual version or - **force** parameter is provided, asks the operator to confirm the resource DLL upgrade and the restart of the related cluster resource.

-force

Force the installation of the HCL Workload Automation 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 HCL Workload Automation to a new major version, such as 10.2.5. This parameter is optional.

hosts=<hostname1,hostname2...>

The host names of the cluster nodes on which you want to update HCL Workload Automation. Host names have to be separated by commas. This parameter is optional. If you do not specify this parameter, HCL Workload Automation 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_name>"

The HCL Workload Automation Resource Instance name as it appears within the cluster group. The default name is `ITWSWorkstation_<domain>_<user>`. This argument is mandatory for **-changeResName**.

"<new_resource_instance_name>"

The new name you want to use for the HCL Workload Automation resource instance. This argument is also mandatory for **-changeResName**.

Examples

About this task

For all the examples described below it is assumed that HCL Workload Automation 10.2.5 has been installed.

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

`mydom`

Is the Windows® User Domain of the HCL Workload Automation user.

`mytwsuser`

Is the HCL Workload Automation 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.

`mydisk`

Is the name of the Physical Disk resource type within the `myresgroup` resource group.

`my shared desktop`

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 HCL Workload Automation in a Windows cluster environment on page 29](#)
- [Example 2: Install and configure the new custom resource for an existing installation of HCL Workload Automation on page 30](#)
- [Example 3: Add a new agent in a cluster environment with HCL Workload Automation already installed on page 30](#)
- [Example 4: Add a custom resource type instance to an existing cluster group on page 31](#)
- [Example 5: Configure HCL Workload Automation in a new joining node of the cluster on page 31](#)
- [Example 6: Deregister HCL Workload Automation on all nodes of the cluster except for the current node on page 31](#)
- [Example 7: Install a new version of the cluster resource DLL into the cluster on page 32](#)
- [Example 8: Force the upgrading/downgrading of the cluster resource DLL into the cluster on page 32](#)
- [Example 9: First installation of domain manager in Windows cluster environment, specifying generic options on page 32](#)
- [Example 10: Upgrading agents in a cluster-aware configuration on page 33](#)
- [Example 11: First installation of HCL Workload Automation in a Windows cluster environment, defining shared desktop on page 33](#)
- [Example 12: First installation of HCL Workload Automation in Windows cluster environment, using customized resource instance name on page 34](#)
- [Example 13: Changing the resource instance name on page 34](#)

Example 1: First installation of HCL Workload Automation in a Windows® cluster environment

About this task

First time installation of HCL Workload Automation in a Windows® cluster environment.

```
twClusterAdm.exe -new domain=MYDOM user=mytwsuser pwd=mytwspwd
-res group=myresgroup ip=myip net=mynetname disk=mydisk opts=lwaOn -dll
```

The command:

- Configures HCL Workload Automation on all the nodes of the cluster.
- Installs the new HCL Workload Automation 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 HCL Workload Automation Cluster resource type within the specified cluster group.
- Adds a dependency from `myip`, `mynetname`, and `mydisk` to the resource.
- Enable monitoring of dynamic agents (`lwaOn`).

Example 2: Install and configure the new custom resource for an existing installation of HCL Workload Automation

About this task

Install and configure the new HCL Workload Automation custom resource for an existing installation of HCL Workload Automation.



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

```
twClusterAdm.exe -new domain=MYDOM user=mytwsuser pwd=mytwspwd
-res group=myresgroup ip=myip net=mynetname disk=mydisk opts=lwaOn -dll
```

The command:

- Installs the new HCL Workload Automation 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 HCL Workload Automation Cluster resource type within the specified cluster group.
- Adds a dependency from `myip`, `mynetname`, and `mydisk` to the resource.
- Enable monitoring of dynamic agents (`lwaOn`).

Example 3: Add a new agent in a cluster environment with HCL Workload Automation already installed

About this task

Add a new HCL Workload Automation agent in a cluster environment where an agent of HCL Workload Automation has been installed and configured in a different Virtual Server.

```
twClusterAdm.exe -new domain=MYDOM user=mytwsuser pwd=mytwspwd
-res group=myresgroup ip=myip net=mynetname disk=mydisk opts=lwaOn
```

The command:

- Configures HCL Workload Automation on all the nodes of the cluster.
- Creates an instance of the HCL Workload Automation Cluster resource type within the specified cluster group
- Adds a dependency from `myip`, `mynetname`, and `mydisk` to the resource.
- Enable monitoring of dynamic agents (IwaOn).

Example 4: Add a custom resource type instance to an existing cluster group

About this task

Add an instance of the HCL Workload Automation custom resource type to an existing cluster group.



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

```
twClusterAdm.exe -new domain=MYDOM user=mytwsuser pwd=mytwspwd
-res group=myresgroup ip=myip net=mynetname disk=mydisk
```

The command:

- Creates an instance of the HCL Workload Automation Cluster resource type within the specified cluster group
- Adds a dependency from `myip`, `mynetname`, and `mydisk` to the resource.

Example 5: Configure HCL Workload Automation in a new joining node of the cluster

About this task

Configure HCL Workload Automation in a new joining node of the cluster.



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

```
twClusterAdm.exe -new domain=MYDOM user=mytwsuser pwd=mytwspwd
hosts=my_new_joining_host_name -dll
```

The command configures HCL Workload Automation and installs the HCL Workload Automation cluster resource DLL on the `my_new_joining_host_name` node.

Example 6: Deregister HCL Workload Automation on all nodes of the cluster except for the current node

About this task

Deregister HCL Workload Automation on all the nodes of the cluster except for the current node. See the section relative to uninstall procedure for more details.

```
twscClusterAdm.exe -uninst domain=MYDOM user=mytwsuser
```

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

To uninstall HCL Workload Automation from the current node you must use the normal uninstallation procedure described in *Planning and Installation Guide*.

Example 7: Install a new version of the cluster resource DLL into the cluster

About this task

Install a new version of the HCL Workload Automation cluster resource DLL into the cluster.

```
twscClusterAdm.exe -update resource=<resource_name>
```

The command upgrades the HCL Workload Automation Cluster resource type if a new version is available.

Example 8: Force the upgrading/downgrading of the cluster resource DLL into the cluster

About this task

Force the upgrading/downgrading of the HCL Workload Automation cluster resource DLL into the cluster.

```
twscClusterAdm.exe -update resource=<resource name> -force
```

The command upgrades the HCL Workload Automation cluster resource without verifying if the version is greater than the version of the installed version.

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

About this task

First time installation of an HCL Workload Automation domain manager in a Windows® cluster environment, specifying the `kill` and `link` generic options

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

The command:

- Configures HCL Workload Automation on all the nodes of the cluster.
- Installs the new HCL Workload Automation 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 HCL Workload Automation Cluster resource type within the specified cluster group.

- Adds a dependency from `myip`, `myname`, and `mydisk` to the resource.
- Sets the generic options `kill` and `link`.

The `link` option specifies 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. 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 HCL Workload Automation 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.

Example 10: Upgrading agents in a cluster-aware configuration

About this task

Upgrade agents in a cluster-aware configuration as follows:

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

```
Suspend-ClusterNode -Name <String>
```

2. Stop all the HCL Workload Automation resources belonging to the nodes of the cluster you paused:

```
Stop-ClusterResource -Name <String>
```

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
-passwd password1
-inst_dir "C:\Program Files\HCL\TWA" -recovInstReg true
```

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

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

- c. Repeat steps [3.a on page 33](#) and [3.b on page 33](#) for all the resources present on this node.
4. From the powershell, resume all the nodes of the cluster you paused in step [1 on page 33](#) by running the following command against each node:

```
Resume-ClusterNode -Name <String>
```

5. Bring online the HCL Workload Automation resources on all the nodes, by running the following command against each resource:

```
Start-ClusterResource -Name <String>
```

Example 11: First installation of HCL Workload Automation in a Windows® cluster environment, defining shared desktop

About this task

First time installation of HCL Workload Automation 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):

```
twscClusterAdm.exe -new domain=MYDOM user=mytwsuser pwd=mytwspwd
-res group=myresgroup ip=myip net=mynetname disk=mydisk opts=lwa0n -dll -sharedesktop
```

The command:

- Configures HCL Workload Automation on all the nodes of the cluster.
- Installs the new HCL Workload Automation 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 HCL Workload Automation Cluster resource type within the specified cluster group.
- Adds a dependency from `myip`, `mynetname`, and `mydisk` to the resource.
- Enable monitoring of dynamic agents (lwaOn).
- Defines that **jobmon** uses the default shared desktop name

Example 12: First installation of HCL Workload Automation in Windows® cluster environment, using customized resource instance name

About this task

First time installation of HCL Workload Automation 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):

```
twscClusterAdm.exe -new domain=MYDOM user=mytwsuser pwd=mytwspwd
-res group=myresgroup ip=myip net=mynetname disk=mydisk resname=myResName opts=lwa0n -dll
```

The command:

- Configures HCL Workload Automation on all the nodes of the cluster.
- Installs the new HCL Workload Automation 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 HCL Workload Automation 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`.
- Enable monitoring of dynamic agents (lwaOn).

Example 13: Changing the resource instance name

About this task

Changing the name of an existing resource instance:

```
twscClusterAdm.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

About this task

First time installation of an HCL Workload Automation domain manager in a Windows® cluster environment, specifying the `lwaOn` generic option.

```
twClusterAdm.exe -new domain=MYDOM user=mytwsuser pwd=mytwspwd -res
group=myresgroup ip=myip net=mynetname disk=mydisk
opts=lwaOn
```

The command:

- Configures HCL Workload Automation on all the nodes of the cluster.
- Installs the new HCL Workload Automation 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 HCL Workload Automation Cluster resource type within the specified cluster group.
- Adds a dependency from `myip`, `mynetname`, and `mydisk` to the resource.
- Sets the generic option `lwaOn` to enable monitoring of the dynamic scheduling agent.

Operating HCL Workload Automation in Windows® cluster environment

About this task

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

- [Cluster resource dependencies on page 35](#)
- [Start up and shut down HCL Workload Automation on page 36](#)
- [The new “cluster instance name” local option on page 37](#)

Cluster resource dependencies

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

To ensure HCL Workload Automation works correctly, the HCL Workload Automation cluster resource instance has to depend on the following resource types:

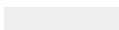
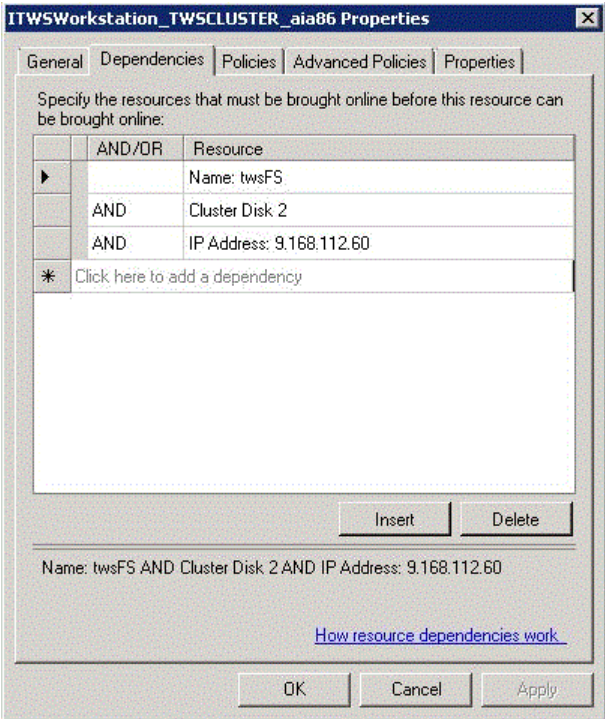
- 

Figure 5. Resource Dependencies tab (Windows Server 2008)



You can decide to add more dependencies to ensure HCL Workload Automation launches jobs only after a given service is available. This happens when HCL Workload Automation 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 HCL Workload Automation cluster resource.

Start up and shut down HCL Workload Automation

About this task

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

- [Redacted]
- StartUp.cmd
- [Redacted]

Use the following scripts to stop and start HCL Workload Automation (you can rename then if required):

- [Redacted]

The above scripts will be automatically created under the HCL Workload Automation installation directory by the `twscClusterAdm.exe` program.

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

Stop

About this task

cluster res <TWSresource instance name> /offline

Start

About this task

cluster res <TWS resource instance name>/online

Examples

About this task

If `ITWSWorkstation_DOMAIN_MST_UserR` is the name of the TWS resource instance, to shut down HCL Workload Automation you have to use:

```
cluster res ITWSWorkstation_DOMAIN_MST_UserR /offline
```

To start HCL Workload Automation 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

About this task

One of the steps of the `twscClusterAdmin` utility is instance name registration of the HCL Workload Automation cluster resource within the local option `localopts` file.

The HCL Workload Automation agent uses the value of this new local option to signal to the HCL Workload Automation 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 HCL Workload Automation resource instance name is changed. If the `clusterinstancename` local option does not point to the right name, the HCL Workload Automation 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 HCL Workload Automation resource instance name use the following procedure:

1. Take the HCL Workload Automation resource instance offline using the Cluster Administrator console. To verify if HCL Workload Automation stopped correctly you can use the Cluster Administrator console and check the status of the HCL Workload Automation resource instance. If the resource has failed to stop you can check in the cluster and HCL Workload Automation logs for the reason. See [Traces on page 40](#) for more details on the log files.
2. From the **Failover Cluster manager** console, select the resource and modify the HCL Workload Automation cluster resource name from `cluster_resource_name` to `cluster_resource_name_new`.



Note: Do not specify this name in double-byte character set (DBCS) characters.

3. Open the `localopts` file using Notepad. Modify the value of the `clusterinstancename` local option. Check that the name is the same you specified for the HCL Workload Automation cluster resource in [Step 2 on page 38](#).
4. Modify the cluster instance name in the `StartUp_clu.cmd` and `ShutDown_clu.cmd` scripts.
5. Bring the HCL Workload Automation resource instance online in the **Failover Cluster manager** console.

HCL Workload Automation Cluster Administrator extension

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

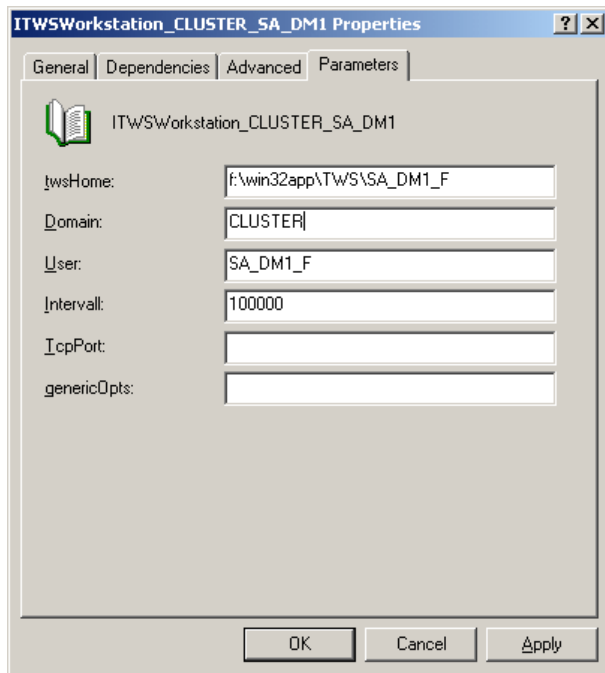
- [Cluster Administrator extension overview on page 38](#)
- [Installing the Cluster Administrator extension on page 39](#)

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 HCL Workload Automation 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 HCL Workload Automation resource properties.

Figure 6. New Properties Parameters tab



The graphic shows the new properties page **Parameters** tab that allows you to modify the `ITWSWorkstation` cluster resource parameters.

Installing the Cluster Administrator extension

About this task

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 6: New Properties Parameters tab on page 39](#)). 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 HCL Workload Automation

About this task

The steps to remove the product must be launched from the same node that was used to install HCL Workload Automation and subsequent fix packs.

Complete the following procedure:

1. Run the utility `TwsClusterAdm -uninst`. This utility removes the HCL Workload Automation 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 the HCL Workload Automation service must be removed. Do not specify the primary node where the HCL Workload Automation agent was installed, otherwise you will remove the original service and make the HCL Workload Automation agent unusable.
2. Manually remove the HCL Workload Automation custom resource instance from the cluster group. You can use the Cluster Administrator console to do this.
3. Optionally, deregister the resource type by using the command:

```
cluster restype ITWSWorkstation /delete
```



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

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

To remove HCL Workload Automation from the current node, you can run the normal uninstallation program, see *Planning and Installation Guide*.

Troubleshooting

This part of the guide gives troubleshooting information about HCL Workload Automation in a Windows® cluster environment. The information here applies to the HCL Workload Automation engine and its installation for this environment. For more troubleshooting information about HCL Workload Automation, see *Troubleshooting Guide*.

For more information, see the following sections:

- [Traces on page 40](#)
- [Error 1314 taking online the resource and the Workstation does not link on page 41](#)
- [HCL Workload Automation resource instance reports fail status or HCL Workload Automation user jobs go in the abend state on page 42](#)
- [Windows Report panel with Jobmon.exe on page 42](#)
- [Cluster: IP validation error on Netman stdlist on page 42](#)

Traces

HCL Workload Automation maintains logs for different activities in different places.

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

clu_offline.log

When the HCL Workload Automation custom resource is taken offline (each time a failover happens), the HCL Workload Automation 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 HCL Workload Automation custom resource is brought online (each time a failover happens), the HCL Workload Automation 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 HCL Workload Automation 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 HCL Workload Automation 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 HCL Workload Automation 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 → 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 in order to enable the Cluster Administrator to act as the HCL Workload Automation user. In this way the HCL Workload Automation custom resource, that runs with the rights of the Cluster Administrator user, is able to stop, start, and link HCL Workload Automation. Reboot the cluster nodes to have this change take effect. This operation is required only the first time you configure HCL Workload Automation to work in the Windows® cluster environment.

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

HCL Workload Automation resource instance reports fail status or HCL Workload Automation user jobs go in the abend state

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

- The HCL Workload Automation cluster resource instance is in fail status.
- HCL Workload Automation 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
+ +++++
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, e: 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 55](#)

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 HCL Workload Automation user. This does not cause a problem because HCL Workload Automation on node B works correctly after a second failover, and HCL Workload Automation 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 HCL Workload Automation is starting and not the IP address of the 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 3. HCL Workload Automation with HACMP

For detailed information, see the following sections:

- [High-Availability Cluster Multi-Processing on page 43](#)
- [UNIX cluster overview on page 46](#)

High-Availability Cluster Multi-Processing

The High-Availability Cluster Multi-Processing (HACMP) tool builds 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 on page 43](#)
- [Physical components of an HACMP cluster on page 44](#)

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

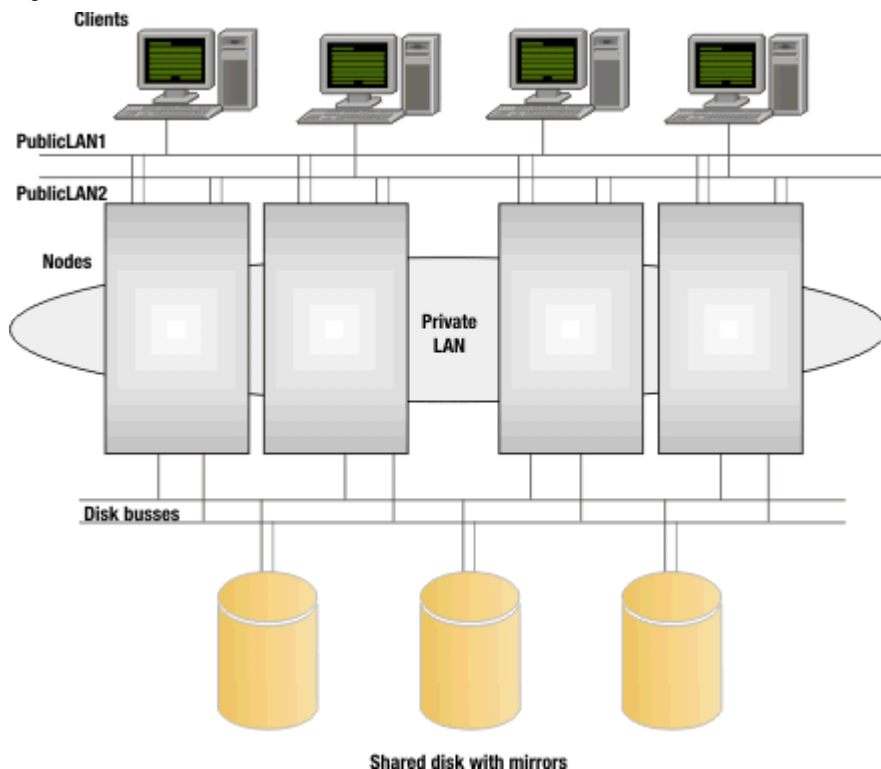
HACMP™ provides a highly-available environment by identifying a set of resources that are essential to uninterrupted processing, and by defining a protocol that nodes use to collaborate to ensure that these resources are available. HACMP™ extends the clustering model by defining relationships among cooperating 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 45](#)
- [Shared external disk devices on page 45](#)
- [Networks on page 45](#)
- [Clients on page 46](#)

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 7: Shared disk with mirror on page 44](#) 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.

Figure 7. Shared disk with mirror



Nodes

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 46](#)
- [Standby and takeover configurations on page 46](#)
- [Design limitations on page 48](#)
- [Supported configurations on page 49](#)

Prerequisite knowledge

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

HCL Workload Automation

For an overview of HCL Workload Automation, see *Overview*.

HACMP clusters

For a Quick Start Guide for HACMP clusters, see AIX Version 7.3 documentation at <https://www.ibm.com/docs/en/aix/7.3>

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.

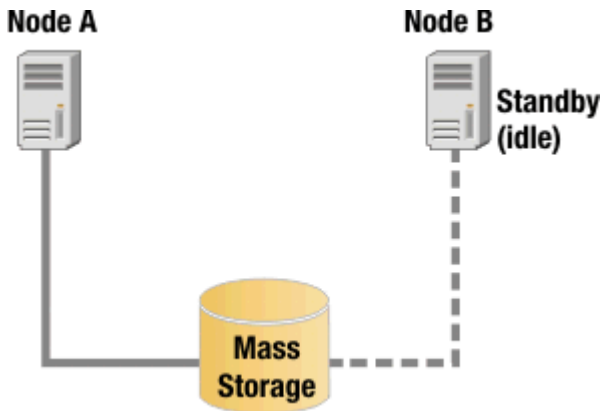
Takeover

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.

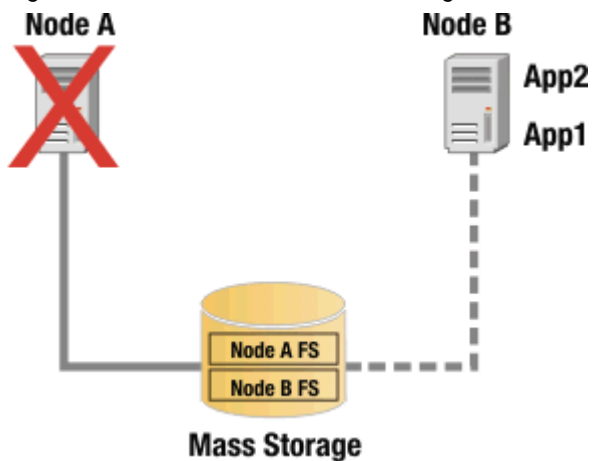
As shown in [Figure 8: Active-Passive configuration in normal operation on page 47](#), Node A is the primary node, and Node B is the standby node currently idling. Although Node B has a connection to the shared mass storage resource, it is not active during normal operation.

Figure 8. Active-Passive configuration in normal operation



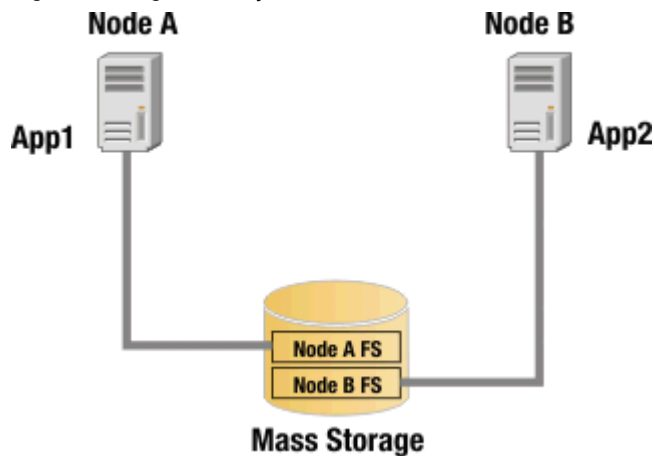
After Node A fail over to Node B, the connection to the mass storage resource from Node B will be activated, and because Node A is unavailable, its connection to the mass storage resource is inactive. This is shown in [Figure 9: Failover on Active-Passive configuration on page 47](#).

Figure 9. Failover on Active-Passive configuration



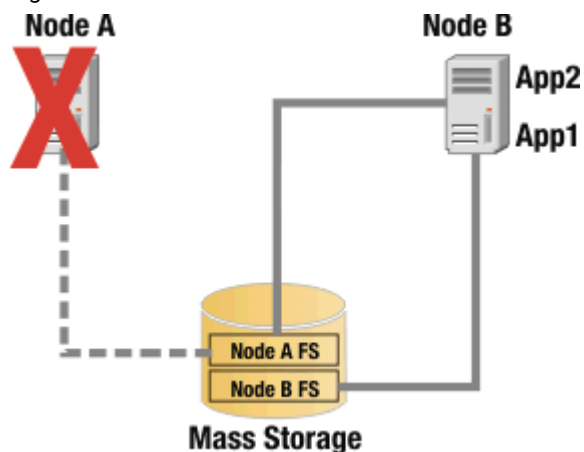
By contrast, in the following a takeover configuration, both Node A and Node B access the shared disk resource simultaneously. For HCL Workload Automation high-availability configurations, this usually means that the shared disk resource has separate, logical file system volumes, each accessed by a different node. This is illustrated in [Figure 10: Logical file system volumes on page 48](#).

Figure 10. Logical file system volumes



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 11: Failover scenario on page 48](#).

Figure 11. Failover scenario



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 on page 49](#)
- [HCL Workload Automation commands on page 49](#)
- [Final status on running jobs on page 49](#)

The master domain manager

The HCL Workload Automation 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 HCL Workload Automation 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 HCL Workload Automation 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 HCL Workload Automation resource) they might corrupt some vital HCL Workload Automation information.

HCL Workload Automation commands

Any HCL Workload Automation command that is running during a failover is not automatically taken offline (unlike the main processes netman, mailman, batchman, and jobman) by the HCL Workload Automation 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 that was set up for the test environment followed by in-depth scenario descriptions.

For the scenarios, the following resources were defined:

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

The Heartbeat on was also configured 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 HCL Workload Automation 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 HCL Workload Automation:

- [Scenario: Shared disk, passive–active failover on a master domain manager on page 50](#)
- [Shared Disk, Passive – Active Failovers on Fault-Tolerant Agent on page 52](#)
- [Switching Domain Managers on page 52](#)

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 HCL Workload Automation 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 HCL Workload Automation and a remote DB2® database

The following procedure explains how to configure HCL Workload Automation 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.

Install HCL Workload Automation using one of the installation methods described in *Planning and Installation Guide*.

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 HCL Workload Automation.

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

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

To install HCL Workload Automation 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 HCL Workload Automation 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 HCL Workload Automation and the remote DB2® server.
5. Proceed with the HCL Workload Automation installation, using `twuser` as the home directory and the local `db2inst1` instance.

After you installed HCL Workload Automation, 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 HCL Workload Automation 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 HCL Workload Automation for HACMP, perform the following steps:

1. Run the cluster collector tool.

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

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

3. Copy `tw_user_home/useropts_twuser` 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. This is described in Changing the workstation host name or IP address. This is described in the topic about changing the workstation host name or IP address in the *Administration Guide*.
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` parameter that is used to run the start and stop of the DB2® instance during the “failover” phase.
7. Try the `start_tws.sh` and `stop_tws.sh` scripts to verify HCL Workload Automation 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 HCL Workload Automation libraries. From the `TWA_home/TWS/bin` directory, run:

```
./twClusterCollector.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 HCL Workload Automation , and link or unlink all the workstations.

Local DB2®

This scenario includes all of the steps described in [Configuring HCL Workload Automation and a remote DB2 database on page 50](#) but, you must also perform the following 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`
3. To stop the monman process used for Event Driven Workload Automation, add "conman startmon" and "conman stopman" to the `start_tws.sh` and `stop_tws.sh` scripts respectively.

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 50](#), but there are no additional steps to perform on the DB2® and WebSphere® Application Server and the `start_tws.sh` and `top_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

You can upgrade cluster nodes to the latest version of the product.

You can upgrade from:

- V9.1 or later

Perform the following actions:

- Read the system requirements.
- Read the chapter on upgrading in *HCL Workload Automation: Planning and Installation*.
- 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.
- Before upgrading an agent in a Windows cluster from Version 9.4, Fix Pack 4 to a later version, ensure that Microsoft Visual Studio C++ redistributable package is manually installed in the second node. For more information, see [Workload Automation Installation /Upgrade and Runtime failure for missing Microsoft Visual C++ Redistributable](#).
- 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 topic about configuring authentication in the *HCL Workload Automation: Administration Guide* to configure your authentication mechanism to use the Federated User Registry.
- The dynamic agent, if installed and 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. .

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 `twscClusterCollector.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 `twscClusterCollector.sh` utility.

After upgrading

Instructions to follow after upgrading.

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 sections about upgrading a master domain manager instance or its backup and upgrading a master domain manager or backup master domain manager instance in *HCL Workload Automation: Planning and Installation*.
- To configure the product to work with the HACMP service IP address, follow the instructions in the topic about changing the workstation host name or IP address in the *HCL Workload Automation: Administration Guide*. 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 A. 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 HCL Workload Automation installed on a workstation in a Windows® 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 on page 55](#)
- [Solutions on page 56](#)
- [Implementing the solutions on page 57](#)

Problem description

The problem occurs because of the way Windows® handles its desktop heap memory, and the way HCL Workload Automation 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 20MB.

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.

HCL Workload Automation processes make the following use of desktops:

HCL Workload Automation Netman Windows® service

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

HCL Workload Automation Token Service Windows® service

Creates a non-interactive desktop for the TWSUser for each agent.

HCL Workload Automation 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

About this task

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

- [Modify the shared heap buffer sizes on page 56](#)
- [Configure the HCL Workload Automation Windows service to start as a local system account on page 57](#)
- [Customize the desktop name so that it is reused on page 57](#)

Modify the shared heap buffer sizes

About this task

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 58](#) for how to do it.

Configure the HCL Workload Automation Windows® service to start as a local system account

About this task

By default, the *HCL Workload Automation 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 10.2.5, 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 on page 58](#) for how to do it.

Customize the desktop name so that it is reused

About this task

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 10.2.5, 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 [twscClusterAdm command with examples of usage on page 21](#) 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 58](#) for how to do it.

Implementing the solutions

About this task

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

- [Modify configuration of Windows service on page 58](#)
- [Modify the Windows registry entries that determine the heap size on page 58](#)
- [Modify localopts to supply a shared desktop name on page 58](#)

Modify configuration of Windows® service

About this task

To modify the *HCL Workload Automation* 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 **HCL Workload Automation** 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

About this task

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 55](#) 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

About this task

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

1. Open the `localopts` file for the agent in question. For the location of this file, see *Planning and Installation Guide*.
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 HCL Workload Automation to make the change effective.

Notices

This document provides information about copyright, trademarks, terms and conditions for product documentation.

© Copyright IBM Corporation 1993, 2016 / © Copyright HCL Technologies Limited 2016, 2025

This information was developed for products and services offered in the US. This material might be available from HCL in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

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

HCL may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

HCL

330 Potrero Ave.

Sunnyvale, CA 94085

USA

Attention: Office of the General Counsel

For license inquiries regarding double-byte character set (DBCS) information, contact the HCL Intellectual Property Department in your country or send inquiries, in writing, to:

HCL

330 Potrero Ave.

Sunnyvale, CA 94085

USA

Attention: Office of the General Counsel

HCL TECHNOLOGIES LTD. PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. HCL 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-HCL 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 HCL product and use of those websites is at your own risk.

HCL may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

HCL

330 Potrero Ave.

Sunnyvale, CA 94085

USA

Attention: Office of the General Counsel

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 HCL under terms of the HCL Customer Agreement, HCL International Program License Agreement or any equivalent agreement between us.

The performance data discussed herein is presented as derived under specific operating conditions. Actual results may vary.

Information concerning non-HCL products was obtained from the suppliers of those products, their published announcements or other publicly available sources. HCL has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-HCL products. Questions on the capabilities of non-HCL products should be addressed to the suppliers of those products.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to HCL, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. HCL, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. HCL shall not be liable for any damages arising out of your use of the sample programs.

© (HCL Technologies Limited) (2025).

Portions of this code are derived from Sample Programs.

© Copyright 2016

Trademarks

HCL®, and other HCL graphics, logos, and service names including "hcltech.com" are trademarks of HCL. Except as specifically permitted herein, these Trademarks may not be used without the prior written permission from HCL. All other trademarks not owned by HCL that appear on this website are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by HCL.

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

IT Infrastructure Library™ is a Registered Trade Mark of AXELOS Limited.

Linear Tape-Open™, LTO™, the LTO™ Logo, Ultrium™, and the Ultrium™ logo are trademarks of HP, IBM® Corp. and Quantum in the U.S. and other countries.

Intel™, Intel™ logo, Intel Inside™, Intel Inside™ logo, Intel Centrino™, Intel Centrino™ logo, Celeron™, Intel Xeon™, Intel SpeedStep™, Itanium™, and Pentium™ are trademarks or registered trademarks of Intel™ Corporation or its subsidiaries in the United States and other countries.

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

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



Java™ and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Cell Broadband Engine™ is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.

ITIL™ is a Registered Trade Mark of AXELOS Limited.

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

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the HCL website.

Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of HCL.

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of HCL.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

HCL reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by HCL, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

HCL MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

Index

Special Characters

- ask, twsClusterAdm.exe argument 27
- changeResName, twsClusterAdm.exe parameter 28
- dll, twsClusterAdm.exe parameter 26
- force, twsClusterAdm.exe argument 28
- force, twsClusterAdm.exe parameter 26
- new, twsClusterAdm.exe parameter 23
- res, twsClusterAdm.exe parameter 24
- sharedDesktop, twsClusterAdm.exe parameter 26
- twsupd, twsClusterAdm.exe argument 28
- uninst, twsClusterAdm.exe parameter 27
- update, twsClusterAdm.exe parameter 27

A

- abend state 42
- accessibility vii
- agent, installing in a cluster-aware configuration 18
- arguments, to twsClusterAdm.exe 23
- ask, twsClusterAdm.exe argument 27

B

- backup domain manager
 - HCL Workload Automation 8

C

- changeResName, twsClusterAdm.exe parameter 28
- check_interval, twsClusterAdm.exe argument 24
- clu_offline.log 41
- clu_online.log 41
- Cluster Administrator console, using 38
- Cluster Administrator extension
 - installing 39
 - overview 38
 - Parameters tab 39
- cluster HACMP upgrading
 - to version 8.6 54
 - to version 9.x 52
- cluster instance name
 - local option 37
 - modifying
 - in the localopts file 38
 - in the ShutDown_clu.cmd script 38
 - in the StartUp_clu.cmd script 38
- cluster resource dependencies 35
 - IP Address 35
 - Network Name 35
 - Physical Disk 35
- cluster.log 41
- commands
 - cluster res 37, 37
 - ShutDown_clu.cmd 36
 - Shutdown.cmd 18
 - StartUp_clu.cmd 36
 - tws_env.cmd 18
 - twsClusterAdm.exe 21
 - twsClusterAdmin.exe, local option 37
- configuration 46
- Conman shut 36
- Conman start 36
- contents, Windows Cluster Enabler 14
- core dump 42
- custom resource DLL 13

- entry-points
 - IsAlive 13
 - Offline 13
 - Online 14
 - Terminate 14

D

- disk, twsClusterAdm.exe argument 24
- DLL
 - custom resource DLL 13
- DLL files
 - ITWSResources.dll 14
 - ITWSWorkstationEx.dll 14
- dll, twsClusterAdm.exe parameter 26
- domain, twsClusterAdm.exe argument 23, 27
- Dynamic Workload Console
 - accessibility vii

E

- error 1314 41
- examples
 - twsClusterAdm.exe 28

F

- failover, twsClusterAdm.exe argument 24
- fault-tolerance
 - definition 8
- force, twsClusterAdm.exe argument 28
- force, twsClusterAdm.exe parameter 26

G

- group, twsClusterAdm.exe argument 24

H

- HACMP cluster upgrading
 - to version 8.6 54
- HCL Workload Automation
 - 45, 45, 46, 46, 49
 - authorization 12
 - backup domain manager 8
 - benefits 43
 - clients 46
 - Cluster Administrator extension 38
 - installing 39
 - overview 38
 - cluster environment
 - integrating into 10
 - cluster resource dependencies 35
 - command-line
 - not automatically taken offline during a failover 11, 49
 - configuring
 - with twsClusterAdm 21
 - HACMP 43, 44, 45
 - in the Windows cluster environments 35
 - integrating into
 - cluster environment 10
 - make cluster-aware 18
 - master domain manager 49
 - not supported on Cluster Virtual Server 11
 - networks 45
 - nodes 45
 - physical components 44
 - product design limitations 11, 48
 - resource instance name
 - changing 38
 - security authentication 12
 - shared external disk 45

- standby 46
- starting 36
 - cluster res command 37
- stopping 36
 - cluster res command 37
- troubleshooting 40
 - abend state 42
 - core dump 42
 - error 1314 41
 - trace files 40
- uninstalling 39
- UNIX 43
- where to find information 10, 46
- Windows 10
- Windows Cluster Enabler 12
 - components 12
- HCL Workload Automation
- agent
 - installing
 - in a cluster-aware configuration 18
- high availability
 - types of 8
- high-availability
 - definition 8
- hosts, twsClusterAdm.exe argument 23, 27, 28

I

- installation
 - installing in a cluster 15
 - prerequisites 16
- installing
 - Cluster Administrator extension 39
 - HCL Workload Automation
 - agent 18
 - Windows Cluster Enabler 14
- installing in a cluster 15
- IP Address 35
- ip, twsClusterAdm.exe argument 24
- Isalive, twsClusterAdm.exe argument 25
- ITWSExInst.cmd 15
- ITWSResources.dll 14
- ITWSWorkstationEx.dll 14

L

- localopts file
 - modifying 38
- lookalive, twsClusterAdm.exe argument 24

M

- master domain manager 49
 - not supported on Cluster Virtual Server reasons why 11

N

- name, twsClusterAdm.exe argument 27
- net, twsClusterAdm.exe argument 24
- Network Name 35
- new_resource_instance_name, twsClusterAdm.exe argument 28
- new, twsClusterAdm.exe parameter 23

O

- operating systems 11
- opts, twsClusterAdm.exe argument 25

P

- Parameters tab
 - Cluster Administrator extension 39
- parameters, to twsClusterAdm.exe 23

- path, twsClusterAdm.exe argument 26
- Physical Disk 35
- prerequisites 16
- pwd, twsClusterAdm.exe argument 23

R

- recovery action in case of failover 8
 - allowing for 8
- Replace a process level token
 - security policy 17, 41
- res, twsClusterAdm.exe parameter 24
- rename, twsClusterAdm.exe argument 24
- resource instance name
 - changing 38
- resource_instance_name, twsClusterAdm.exe argument 28
- resource, twsClusterAdm.exe argument 27
- roll back or recovery action 8
 - allowing for 8

S

- scripts
 - ShutDown_clu.cmd 36
 - StartUp_clu.cmd 36
- security policy
 - Replace a process level token 17, 41
- sharedDesktop, twsClusterAdm.exe parameter 26
- ShutDown_clu.cmd 36
- Shutdown.cmd 18, 36
- ShutdownLwa.cmd 36
- StartUp_clu.cmd 36
- StartUp.cmd 36
- StartupLwa.cmd 36
- syntax
 - twsClusterAdm.exe 22

T

- takeover 46
- tcpport, twsClusterAdm.exe argument 25
- to version 9.x
 - cluster HACMP upgrading to 52
- trace files 40
 - clu_offline.log 41
 - clu_online.log 41
 - cluster.log 41
- troubleshooting 40
 - abend state 42
 - error 1314 41
 - Jobmon
 - core dump 42
 - trace files 40
- tws_env.cmd 18
- twsClusterAdm
 - HCL Workload Automation agent
 - 21
- twsClusterAdm.exe 14, 18, 21
 - arguments 23
 - examples 28
 - parameters 23
 - syntax 22
- twsClusterAdmin.exe
 - cluster instance name
 - local option 37
- twshome, twsClusterAdm.exe argument 23
- twsupd, twsClusterAdm.exe argument 28

U

- uninst, twsClusterAdm.exe parameter 27
- UNIX 43
 - where to find information 46

- update, twsClusterAdm.exe parameter 27
- upgrading
 - Workload Scheduler agent 20
- upgrading in HACMP cluster
 - to version 8.6 54
 - to version 9.x 52
- upgrading manually 20
- user, twsClusterAdm.exe argument 23, 27
- utility
 - Shutdown.cmd 18

V

- version 8.6
 - cluster HACMP upgrading to 54

W

- Windows 10
- Windows Cluster Enabler
 - components 12
 - custom resource DLL 13
 - contents 14
 - ITWSExInst.cmd 15
 - ITWSResources.dll 14
 - ITWSWorkstationEx.dll 14
 - twsClusterAdm.exe 14
 - installing 14
 - installing in a cluster 15
 - prerequisites 16
- Windows clusters
 - where to find information 10
- Workload Scheduler agent
 - upgrading manually
 - in a cluster-aware configuration 20