Veritas™ High Availability Agent for WebSphere Application Server Installation and Configuration Guide

AIX, HP-UX, Linux, Solaris

5.1

June 2012

Veritas High Availability Agent for WebSphere Application Server Installation and Configuration Guide

The software described in this book is furnished under a license agreement and may be used only in accordance with the terms of the agreement.

Agent Version: 5.1.10.0

Document version: 5.1.10.0.1

Legal Notice

THE DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID. SYMANTEC CORPORATION SHALL NOT BE LIABLE FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS DOCUMENTATION. THE INFORMATION CONTAINED IN THIS DOCUMENTATION IS SUBJECT TO CHANGE WITHOUT NOTICE.

The Licensed Software and Documentation are deemed to be commercial computer software as defined in FAR 12.212 and subject to restricted rights as defined in FAR Section 52.227-19 "Commercial Computer Software - Restricted Rights" and DFARS 227.7202, "Rights in Commercial Computer Software or Commercial Computer Software Documentation", as applicable, and any successor regulations. Any use, modification, reproduction release, performance, display or disclosure of the Licensed Software and Documentation by the U.S. Government shall be solely in accordance with the terms of this Agreement.

Symantec Corporation
350 Ellis Street
Mountain View, CA 94043

http://www.symantec.com

Technical Support

Symantec Technical Support maintains support centers globally. Technical Support's primary role is to respond to specific queries about product features and functionality. The Technical Support group also creates content for our online Knowledge Base. The Technical Support group works collaboratively with the other functional areas within Symantec to answer your questions in a timely fashion. For example, the Technical Support group works with Product Engineering and Symantec Security Response to provide alerting services and virus definition updates.

Symantec's support offerings include the following:

- A range of support options that give you the flexibility to select the right amount of service for any size organization
- Telephone and/or Web-based support that provides rapid response and up-to-the-minute information
- Upgrade assurance that delivers software upgrades
- Global support purchased on a regional business hours or 24 hours a day, 7 days a week basis
- Premium service offerings that include Account Management Services

For information about Symantec's support offerings, you can visit our Web site at the following URL:

www.symantec.com/business/support/index.jsp

All support services will be delivered in accordance with your support agreement and the then-current enterprise technical support policy.

Contacting Technical Support

Customers with a current support agreement may access Technical Support information at the following URL:

www.symantec.com/business/support/contact_techsupp_static.jsp

Before contacting Technical Support, make sure you have satisfied the system requirements that are listed in your product documentation. Also, you should be at the computer on which the problem occurred, in case it is necessary to replicate the problem.

When you contact Technical Support, please have the following information available:

- Product release level

- Hardware information
- Available memory, disk space, and NIC information
- Operating system
- Version and patch level
- Network topology
- Router, gateway, and IP address information
- Problem description:
  - Error messages and log files
  - Troubleshooting that was performed before contacting Symantec
  - Recent software configuration changes and network changes

Licensing and registration

If your Symantec product requires registration or a license key, access our technical support Web page at the following URL:

www.symantec.com/business/support/

Customer service

Customer service information is available at the following URL:

www.symantec.com/business/support/

Customer Service is available to assist with non-technical questions, such as the following types of issues:

- Questions regarding product licensing or serialization
- Product registration updates, such as address or name changes
- General product information (features, language availability, local dealers)
- Latest information about product updates and upgrades
- Information about upgrade assurance and support contracts
- Information about the Symantec Buying Programs
- Advice about Symantec's technical support options
- Nontechnical presales questions
- Issues that are related to CD-ROMs or manuals

Support agreement resources

If you want to contact Symantec regarding an existing support agreement, please contact the support agreement administration team for your region as follows:

Asia-Pacific and Japan	customercare_apac@symantec.com
Europe, Middle-East, and Africa	semea@symantec.com
North America and Latin America	supportsolutions@symantec.com

Documentation

Product guides are available on the media in PDF format. Make sure that you are using the current version of the documentation. The document version appears on page 2 of each guide. The latest product documentation is available on the Symantec Web site.

https://sort.symantec.com/documents

Your feedback on product documentation is important to us. Send suggestions for improvements and reports on errors or omissions. Include the title and document version (located on the second page), and chapter and section titles of the text on which you are reporting. Send feedback to:

doc_feedback@symantec.com

About Symantec Connect

Symantec Connect is the peer-to-peer technical community site for Symantec's enterprise customers. Participants can connect and share information with other product users, including creating forum posts, articles, videos, downloads, blogs and suggesting ideas, as well as interact with Symantec product teams and Technical Support. Content is rated by the community, and members receive reward points for their contributions.

http://www.symantec.com/connect/storage-management

Contents

Chapter 1

Introducing the Veritas High Availability Agent for WebSphere Application Server

This chapter includes the following topics:

About the Veritas agent for WebSphere Application Server

The Veritas High Availability agents monitor specific resources within an enterprise application. They determine the status of resources and start or stop them according to external events.

The Veritas agent for WebSphere Application Server provides high availability for WebSphere Application Server in a clustered environment.

The agent supports the following types of WebSphere Application Server instances:

- Deployment Manager
- Node Agent

- Application Server

See the following Technical Support TechNote for the latest updates or software issues for this agent:

http://www.symantec.com/business/support/index?page=content&id=TECH46455

What's new in this agent

The enhancements in this release of the Websphere Application Server agent are as follows:

- Added support for WebSphere Application Server 8.0.
- Fixed an issue that prevented the WebSphere Application Server agent from monitoring a resource due to the length limit of the process command line.

For information on the changes introduced in the previous releases:

See "Changes introduced in previous releases" on page 71.

Supported software

For information on the software versions that the agent for WebSphere Application Server supports, see the Symantec Operations Readiness Tools (SORT) site: https://sort.symantec.com/agents.

WebSphere Application Server agent functions

The agent consists of resource type declarations and agent executables. The agent executables are organized into online, offline, monitor, and clean functions.

Online

The online function is responsible for starting a WebSphere Application Server. The online function performs the following tasks:

- Verifies that the WebSphere Application Server instance is not already online.
- Determines the version of the WebSphere Application Server software.
- Starts the WebSphere Application Server instance by executing the appropriate start script, which is supplied by the WebSphere installation program. The script executed depends upon the type of server being started.

Server Type	Start Command

Deployment Manager	`WAS_HOME/binDir/startManager.sh`
Node Agent	`WAS_HOME/binDir/startNode.sh`
Application Server	`WAS_HOME/binDir/startServer.sh`

Offline

The offline function is responsible for stopping a WebSphere Application Server instance. The offline function performs the following tasks:

- Verifies that the WebSphere Application Server instance is not already offline.
- Determines the version of the WebSphere Application Server software.
- Stops the WebSphere Application Server instance by executing the appropriate stop script, which is supplied by the WebSphere installation program. The script executed depends upon the type of server being stopped.

Server Type	Stop Command
Deployment Manager	`WAS_HOME/binDir/stopManager.sh`
Node Agent	`WAS_HOME/binDir/stopNode.sh`
Application Server	`WAS_HOME/binDir/stopServer.sh`

Monitor

The monitor function is responsible for monitoring the state of WebSphere Application Servers on all nodes in the cluster.

The monitor function performs the following tasks:

- First-level monitoring quickly checks for the existence of the system process (the Java Virtual Machine) that represents the WebSphere Application Server instance. It determines the process existence by scanning the system process table and searching for strings in the process command line that uniquely identify the JVM process associated with the WebSphere Application Server instance. These search strings include the values specified in resource attributes WAS_HOME, WAS_NODE, and ServerName.
  The agent also supports Intelligent Monitoring Framework (IMF) in the first-level check. IMF enables intelligent resource monitoring. See "How the agent supports intelligent resource monitoring" on page 38. You can use the MonitorFreq key of the IMF attribute to specify the frequency at which the agent invokes the monitor function. See "MonitorFreq" on page 40.

- If second-level monitoring is enabled (if SecondLevelMonitor > 0), the monitor function performs a deeper, more thorough state check of the WebSphere Application Server. Second-level monitoring uses the IBM-supplied utility program serverStatus.sh. The output from this program is parsed to confirm the server is running.

  When enabled, the integer value specified in attribute SecondLevelMonitor determines how frequently the program is executed. For example, if SecondLevelMonitor is set to 1, the monitor function executes serverStatus.sh during each monitor interval. If SecondLevelMonitor is set to 3, the monitor function executes serverStatus.sh every third monitor interval. This mechanism lets you control the system load generated by monitoring.

  The serverStatus.sh script spawns a Java program that establishes a connection to the WebSphere Application Server. Spawning a JVM every monitor interval places additional load on the system. If performance is more important than a second-level state check, then consider disabling second-level monitoring and only performing the first-level process check.

  Note: The attribute used to configure the second-level check and its frequency depends on the software versions of VCS and WebSphere agent you have installed: For VCS 5.1 SP1 or later with Websphere agent version 5.1.9.0, use the LevelTwoMonitorFreq attribute. For VCS 5.1 or earlier with WebSphere agent 5.1.8.0 or earlier, use the SecondLevelMonitor attribute.

- The monitor function executes a custom monitor program specified in the attribute MonitorProgram. This program does not execute if either the first- or second-level monitor reports that the resource is offline. You can omit second-level monitoring, and attempt running a custom monitor check immediately after first-level monitoring.

  This feature allows VCS dministrator to define custom programs that determine the state of the WebSphere Application Server. For example, the administrator may want to test the status of a J2EE component running inside the server and ensure that the underlying application is functioning properly.

  See "WebSphere Application Server agent attributes" on page 28.

Clean

The clean function removes any WebSphere Application Server instance processes remaining after a fault event or after an unsuccessful attempt to online or offline the resource.

The clean function performs the following tasks:

- Kills the process that starts the WebSphere Application Server instance. It is unlikely that this process exists, but it needs to be removed if for some reason it still exists during clean.
- Kills the process that stops the WebSphere Application Server instance. It is unlikely this process exists, but it needs to be removed if for some reason it still exists during clean.
- Kills the JVM process for the WebSphere Application Server instance. This process is identified by searching the system process table using the values specified in attributes WAS_HOME, WAS_NODE, and ServerName.

Note: For information about the additional functions of the agent for WebSphere Application Server when IMF is enabled: See “Agent functions for the IMF functionality” on page 39.

Chapter 2

Installing, upgrading, and removing the agent for WebSphere Application Server

This chapter includes the following topics:

Before you install the Veritas agent for WebSphere Application Server

You must install the Veritas agent for WebSphere Application Server on all the systems that will host WebSphere Application Server service groups.

Ensure that you meet the following prerequisites to install the agent for WebSphere Application Server.

For VCS, do the following:

- Install and configure Veritas Cluster Server.
  For more information on installing and configuring Veritas Cluster Server, refer to the *Veritas Cluster Server Installation Guide*.
- Remove any previous version of this agent.
  To remove the agent,
  See “Removing the agent in a VCS environment” on page 21.
- Install the latest version of ACC Library.
  To install or update the ACC Library package, locate the library and related documentation in the Agent Pack tarball:
  See “Installing the ACC library” on page 19.

Prerequisites for enabling i18n support

Perform the following steps to enable i18n support to the agent:

- Install ACCLib version 5.1.2.0 or later.
  See “Installing the ACC library” on page 19.
- For VCS 5.0 and earlier releases, copy the latest ag_i18n_inc.pm module from the following location on the agent pack disc.

Note: Review the readme.txt for instructions to copy this module.

VCS 5.0	cd1/*platform*/*arch_dist*/vcs/application/i18n_support/5.0
VCS 4.1	cd1/*platform*/*arch_dist*/vcs/application/i18n_support/4.1
VCS 4.0	cd1/*platform*/*arch_dist*/vcs/application/i18n_support/4.0

where *arch_dist* takes the following values:
'sol_sparc' for Solaris SPARC
'sol_x64' for Solaris x64
'generic' for HP-UX and Linux

Note: *arch_dist* is not applicable to AIX.

About the ACC library

The operations of a VCS agent depend on a set of Perl modules known as the ACC library. The library must be installed on each system in the cluster that runs the agent. The ACC library contains common, reusable functions that perform tasks, such as process identification, logging, and system calls.

Instructions to install or remove the ACC library on a single system in the cluster are given in the following sections. The instructions assume that the agent's tar file has already been extracted.

Installing the ACC library

Install the ACC library on each system in the cluster that runs an agent that depends on the ACC library.

To install the ACC library

1. Log in as superuser.
2. Download ACC Library.

   You can download either the complete Agent Pack tarball or the individual ACCLib tarball from the Symantec Operations Readiness Tools (SORT) site (https://sort.symantec.com/agents).
3. If you downloaded the complete Agent Pack tarball, navigate to the directory containing the package for the platform running in your environment.

AIX	*cd1*/aix/vcs/application/acc_library/*version*_library/pkgs
HP-UX	*cd1*/hpux/generic/vcs/application/acc_library/*version*_library/pkgs
Linux	*cd1*/linux/generic/vcs/application/acc_library/*version*_library/rpms
Solaris	*cd1*/solaris/*dist_arch*/vcs/application/acc_library/*version*_library/pkgs where *dist_arch* is sol_sparc or sol_x64.

4. If you downloaded the individual ACCLib tarball, navigate to the pkgs directory (for AIX, HP-UX, and Solaris), or rpms directory (for Linux).
5. Install the package. Enter **Yes** if asked to confirm overwriting of files in the existing package.

AIX	`# installp -ac -d VRTSacclib.bff VRTSacclib`
HP-UX	`# swinstall -s ‘pwd‘ VRTSacclib`

Linux	`# rpm -i \`
	`VRTSacclib-`*`VersionNumber`*`-GA_GENERIC.noarch.rpm`

Solaris	`# pkgadd -d VRTSacclib.pkg`

Installing the agent in a VCS environment

Install the agent for WebSphere Application Server on each node in the cluster.

To install the agent in a VCS environment

1. Download the agent from the Symantec Operations Readiness Tools (SORT) site: https://sort.symantec.com/agents.

   You can download either the complete Agent Pack tar file or an individual agent tar file.
2. Uncompress the file to a temporary location, say /tmp.
3. If you downloaded the complete Agent Pack tar file, navigate to the directory containing the package for the platform running in your environment.

AIX	`cd1/aix/vcs/application/websphere_agent/`
	`vcs_version``/`*`version`*`_agent/pkgs`

HP-UX	`cd1/hpux/generic/vcs/application/websphere_agent/`
	`vcs_version``/`*`version`*`_agent/pkgs`

Linux	`cd1/linux/generic/vcs/application/websphere_agent/`
	`vcs_version``/`*`version`*`_agent/rpms`

Solaris	`cd1/solaris/`*`dist_arch`*`/vcs/application/websphere_agent/`
	`vcs_version``/`*`version`*`_agent/pkgs`
	where, *dist_arch* is sol_x64 or sol_sparc

   If you downloaded the individual agent tar file, navigate to the pkgs directory (for AIX, HP-UX, and Solaris), or rpms directory (for Linux).

4. Log in as superuser.

5. Install the package.

AIX	`# installp -ac -d VRTSvcswas.rte.bff VRTSvcswas.rte`
HP-UX	`# swinstall -s ‘pwd‘ VRTSvcswas`

Linux	`# rpm -ihv \VRTSvcswas-AgentVersion`
	`-GA_GENERIC.noarch.rpm`

Solaris	`# pkgadd -d . VRTSvcswas`

6. After installing the agent package, you must import the agent type configuration file. See “Importing the agent types files in a VCS environment” on page 26.

Removing the agent in a VCS environment

You must uninstall the agent for WebSphere Application Server from a cluster while the cluster is active.

To uninstall the agent in a VCS environment

1. Log in as a superuser.

2. Set the cluster configuration mode to read/write by typing the following command from any node in the cluster:

   ```
   # haconf -makerw
   ```

3. Remove all WebSphere Application Server resources from the cluster. Use the following command to verify that all resources have been removed:

   ```
   # hares -list Type=WebSphere
   ```

4 Remove the agent type from the cluster configuration by typing the following command from any node in the cluster:

```
# hatype -delete WebSphere
```

Removing the agent’s type file from the cluster removes the include statement for the agent from the main.cf file, but the agent's type file is not removed from the cluster configuration directory. You can remove the agent’s type file later from the cluster configuration directory.

5 Save these changes. Then set the cluster configuration mode to read-only by typing the following command from any node in the cluster:

```
# haconf -dump -makero
```

6 Use the platform's native software management program to remove the agent for WebSphere Application Server from each node in the cluster.

Execute the following command to uninstall the agent:

AIX	`# installp -u VRTSvcswas.rte`
HP-UX	`# swremove VRTSvcswas`
Linux	`# rpm -e VRTSvcswas`
Solaris	`# pkgrm VRTSvcswas`

Removing the ACC library

Perform the following steps to remove the ACC library.

To remove the ACC library

1 Ensure that all agents that use ACC library are removed.

2 Run the following command to remove the ACC library package.

AIX	`# installp -u VRTSacclib`
HP-UX	`# swremove VRTSacclib`
Linux	`# rpm -e VRTSacclib`
Solaris	`# pkgrm VRTSacclib`

Updating the WebSphere agent to latest release

Perform the following steps to upgrade the agent with minimal disruption, in a VCS environment

1 Persistently freeze the service groups that host the application.

```
# hagrp -freeze group -persistent
```

2 Stop the cluster services forcibly.

```
# hastop -all -force
```

3 Ensure that the agent operations are stopped on all the nodes.

```
# ps -ef |grep WebSphere
```

4 Uninstall the agent package from all the nodes.

See "To uninstall the agent in a VCS environment" on page 21.

5 Install the new agent on all the nodes.

See "To install the agent in a VCS environment" on page 20.

6 Copy the new WebSphereTypes.cf file from the agent's conf directory, to the VCS conf directory /etc/VRTSvcs/conf/config.

VCS Version	Operating system	Agent types files
VCS 4.x	■ AIX ■ HP-UX ■ Linux ■ Solaris	/etc/VRTSvcs/conf/sample_WebSphere/WebSphere.cf
VCS 5.x	■ AIX ■ HP-UX ■ Linux	/etc/VRTSagents/ha/conf/WebSphere/WebSphereTypes.cf
VCS 5.0	Solaris SPARC and x64	/etc/VRTSagents/ha/conf/WebSphere/WebSphereTypes50.cf
VCS 5.1	Solaris SPARC and x64	/etc/VRTSagents/ha/conf/WebSphere/WebSphereTypes51.cf

7 Check for the changes in the resource values required, if any, due to the new agent types file.

8 Start the cluster services.

```
# haagent -start WebSphere -sys System
```

9 Unfreeze the service groups once all the resources come to an online steady state.

```
# hagrp -unfreeze GroupName -persistent
```

Chapter 3

Configuring the agent for WebSphere Application Server

This chapter includes the following topics:

About configuring the Veritas agent for WebSphere Application Server

After installing the Veritas agent for WebSphere Application Server, you must import the agent type configuration file. After importing this file, you can create and configure WebSphere Application Server resources. Before you configure a resource, review the attributes table that describes the resource type and its attributes.

To view the sample agent type definition and service groups configuration.

See “About sample configurations for the agent for WebSphere Application Server” on page 63.

Importing the agent types files in a VCS environment

To use the agent for WebSphere Application Server, you must import the agent types file into the cluster.

You can import the agent types file using the Veritas Cluster Server (VCS) graphical user interface or via the command line interface.

To import the agent types file using the VCS graphical user interface

1 Start the Veritas Cluster Manager and connect to the cluster on which the agent is installed.

2 Click **File > Import Types**.

3 In the Import Types dialog box, select the following file:

VCS version	Operating System	Agent types file
VCS 4.x	■ AIX ■ HP-UX ■ Linux ■ Solaris	`/etc/VRTSvcs/conf/sample_WebSphere/` `WebSphereTypes.cf`
VCS 5.x or later	■ AIX ■ HP-UX ■ Linux	`/etc/VRTSagents/ha/conf/WebSphere/` `WebSphereTypes.cf`
VCS 5.0	■ Solaris SPARC and x64	`/etc/VRTSagents/ha/conf/WebSphere/` `WebSphereTypes50.cf`
VCS 5.1 or later	■ Solaris SPARC and x64	`/etc/VRTSagents/ha/conf/WebSphere/` `WebSphereTypes51.cf`

4 Click **Import**.

5 Save the VCS configuration.

The WebSphere Application Server agent type is now imported to the VCS engine.

You can now create WebSphere Application Server resources. For additional information about using the VCS GUI, refer to the *Veritas Cluster Server Administrator's Guide*.

To import the agent types file using the Veritas Cluster Server command line interface (CLI):

1. Log on to any one of the systems in the cluster as the superuser.
2. Create a temporary directory.

```
# mkdir ./temp
```

```
# cd ./temp
```

3. Copy the sample file Types.cf.

VCS 4.x	AIX HP-UX Linux Solaris	`/etc/VRTSvcs/conf/sample_WebSphere/WebSphereTypes.cf`
VCS 5.x or later	AIX HP-UX Linux	`/etc/VRTSagents/ha/conf/WebSphere/WebSphereTypes.cf`
VCS 5.0	Solaris SPARC and x64	`/etc/VRTSagents/ha/conf/WebSphere/WebSphereTypes50.cf`
VCS 5.1 or later	Solaris SPARC and x64	`/etc/VRTSagents/ha/conf/WebSphere/WebSphereTypes51.cf`

4. Create a dummy main.cf file:

```
# echo 'include "WebSphereTypes.cf"' > main.cf
```

5. Create the WebSphere resource type as follows:

```
# hacf -verify .
```

```
# haconf -makerw
```

```
# sh main.cmd
```

```
# haconf -dump
```

The WebSphere Application Server agent type is now imported to the VCS engine.

You can now create WebSphere Application Server resources. For additional information about using the VCS CLI, refer to the *Veritas Cluster Server Administrator's Guide*.

WebSphere Application Server agent attributes

Table 3-1 shows the required attributes for the agent for WebSphere Application Server.

Table 3-1 Required attributes

Required attribute	Description
ResLogLevel	The logging detail performed by the agent for the resource. Valid values are: ERROR: Only logs error messages. WARN: Logs above plus warning messages. INFO: Logs above plus informational messages. TRACE: Logs above plus trace messages. TRACE is very verbose and should only be used during initial configuration or for troubleshooting and diagnostic functions. Type and dimension: string-scalar Default: INFO Example: TRACE
ServerName	Contains the server name assigned to the WebSphere Server during its installation. In Network Deployment configurations, the default ServerName for Deployment Managers is dmgr and the default ServerName for the Node Agents is nodeagent, but these names are not mandatory. See "Uniquely identifying WebSphere Application Server instances" on page 32. Type and dimension: string-scalar Default: "" Example: server1
ServerProfile	Server profile name of the WebSphere Server instance or complete path to the WebSphere Application Server profile. This attribute is applicable to WebSphere version 6.0 and later, and must be null if the WebSphere major version number is 5. You must specify this attribute if the resource manages a WebSphere Application Server version 6.0 and later. Type and dimension: string-scalar Default: "" Example 1: Dmgr01 Example 2: /WAS/AppSrv/profiles/AppSrv01/

Table 3-1 Required attributes *(continued)*

Required attribute	Description
ServerType	Type of WebSphere Application Server that the cluster will manage. Valid names are as follows: ■ DeploymentManager: Resource is a Deployment Manager. ■ NodeAgent: Resource is a Node Agent. ■ ApplicationServer: Resource is an Application Server, which may be a stand-alone server or may be part of a Network Deployment and is a member of a WebSphere Cell. The agent uses this value to determine how to manage the WebSphere Application Server within a cluster. Refer to the WebSphere documentation for a full explanation of the purposes and use of each WebSphere Application Server type. Type and dimension: string-scalar Default: "" Example: DeploymentManager
User	The UNIX user name used to run the programs that start, stop, and monitor the WebSphere resource, which include the program specified in the MonitorProgram attribute. IBM recommends using the root account, but you may use any account. If User is not set to root, the user name must be synchronized across the systems within the cluster. In other words, the user name must resolve to the same UID and have the same default shell on each system in the cluster. Type and dimension: string-scalar Default: "" Example: root
WAS_HOME	The absolute path to the WebSphere Application Server or WebSphere Application Server Network Deployment root installation directory. This attribute is used to locate programs executed by the agent. It is also where the *binDir*/setupCmdLine.sh file resides. The value is also used to uniquely identify the ServerType processes. Using WAS_HOME to uniquely identify an Application Server's process IDs requires that WAS_HOME be unique compared to WAS_HOME for all other WAS instances in the cluster. See "Uniquely identifying WebSphere Application Server instances" on page 32. **Note:** Both WAS_HOME and WAS_ND_HOME are defined as WAS_HOME in the standard environment file setupCmdLine.sh, which is supplied with WebSphere. Type and dimension: string-scalar Default: "" Example: /ibm/was/v51/cell1/node2

Table 3-1 Required attributes *(continued)*

Required attribute	Description
WAS_NODE	The WebSphere Node Name to which the server instance belongs. The Node Name is an administrative identifier that is internal to the WebSphere environment and is assigned when the node is installed. WebSphere requires that a Node Name must be unique within a WebSphere cell. See "Uniquely identifying WebSphere Application Server instances" on page 32. Type and dimension: string-scalar Default: "" Example: was51c1n2

Table 3-2 lists the optional attributes for the agent for WebSphere Application Server.

Table 3-2 Optional attributes

Optional Attribute	Definition
MonitorProgram	The full pathname and command-line arguments for an externally-provided custom monitor program. The program is executed within the security context of the UNIX account specified in attribute User. The program must be completely self-contained and independent, and it must return one of the following exit codes: 110 or 0: The WebSphere Application Server is ONLINE. 100 or 1: The WebSphere Application Server is OFFLINE. All other: The WebSphere Application Server state is UNKNOWN. Symantec recommends storing the external monitor program on the shared storage device, in the directory specified by the WAS_HOME attribute, to ensure the file is always available on the online system. Type and dimension: string-scalar Default: "" Example: /usr/WAS51/server1/bin/mymonitor.sh

Table 3-2 Optional attributes *(continued)*

Optional Attribute	Definition
SecondLevelMonitor	Specifies if second-level monitor is enabled and how frequently it is performed. Second-level monitor is a deeper, more thorough state check of the WebSphere resource, performed by executing the IBM-supplied utility program serverStatus.sh. The output from this program is parsed to confirm the server status is running. The integer value specified by this attribute determines how frequently the second-level monitor program is executed. For example, if SecondLevelMonitor is set to 1, the monitor function will execute serverStatus.sh during each monitor interval. A value of 3 executes the program every third monitor interval. If SecondLevelMonitor is set to 0, the monitor function will never perform the second-level monitor. **Note:** The SecondLevelMonitor attribute is applicable to VCS versions earlier than VCS 5.1 SP1 with WebSphere agent versions earlier than 5.1.9.0. From VCS version 5.1 SP1 with WebSphere agent version 5.1.9.0 onwards, the SecondLevelMonitor attribute of the WebSphere agent is deprecated. Instead, a resource type level attribute LevelTwoMonitorFreq should be used to specify the frequency of in-depth monitoring. Type and dimension: integer-scalar Default: 0 Example: 1
LevelTwoMonitorFreq	Specifies the frequency at which the agent for this resource type must perform second-level or detailed monitoring. You can also override the value of this attribute at the resource level. The value indicates the number of monitor cycles after which the agent will monitor WebSphere in detail. For example, the value 5 indicates that the agent will monitor WebSphere in detail after every five online monitor intervals. **Note:** This attribute is applicable to VCS version 5.1 SP1 with WebSphere agent version 5.1.9.0 or later. If the VCS version is earlier than VCS 5.1 SP1 and the WebSphere agent version is earlier than 5.1.9.0, the SecondLevelMonitor attribute should be used. If you upgraded the VCS version to VCS 5.1 SP1 and the WebSphere agent version to 5.1.9.0 (or later), and if you had enabled detail monitoring in the previous version, then do the following: ■ Set the value of the LevelTwoMonitorFreq attribute to the same value as that of the SecondLevelMonitor attribute. Type and dimension: integer-scalar Default: 0

Table 3-2 Optional attributes *(continued)*

Optional Attribute	Definition
StartOptions	The command-line options that are passed to the WebSphere start script when it is executed within the online function. Multiple options should be separated by a space. Refer to the WebSphere product documentation for a list and description of supported start options. Type and dimension: string-scalar Default: "" Example: "-replacelog -trace"
StopOptions	The command-line options that are passed to the WebSphere stop script when it is executed within the offline function. Multiple options should be separated by a space. Refer to the WebSphere product documentation for a list and description of supported stop options. Type and dimension: string-scalar Default: "" Example: "-replacelog -trace"

Note: For information about the additional attributes of the agent for WebSphere Application Server when IMF is enabled: See "Attributes that enable IMF" on page 39.

Uniquely identifying WebSphere Application Server instances

You can virtualize a WebSphere Application Server instance using a cluster. Using shared disk and virtual IP addresses, you can manage a large set of WebSphere Application Server instances in a single cluster.

Set the WAS_HOME, WAS_NODE, and ServerName attributes such that the combined values are unique for each WebSphere Application Server instance.

WebSphere Application Servers can run on separate cluster nodes or can run concurrently on a single node. If WebSphere Application Servers run concurrently on a single node, you must ensure that the agent can uniquely identify each WebSphere Application Server on a host system that is running more than one WebSphere Application Server.

For unique identification, the agent's monitor and clean functions use the values specified by attributes WAS_HOME, WAS_NODE, and ServerName to uniquely identify each running WebSphere Server JVM process.

Differentiating WebSphere Application Server instances is especially important when the agent must kill the processes of a non-responsive or failed instance. Failure to define unique names for each WebSphere Application Server could result in a clean operation that kills processes for more than one WebSphere Application Server instance.

Important considerations while configuring the agent

While configuring the agent, make the following settings:

- The time required to fully start a WebSphere instance depends on the number, size, and complexity of Java applications started within the server. Be sure to compare the value of the OnlineTimeout attribute with the actual time required to fully initialize the WebSphere Application Server. Large WebSphere Application Server deployments may require a larger OnlineTimeout. Properly tuning this attribute ensures that the cluster does not time out the online entry point while a WebSphere Application Server is initializing.
- Allow sufficient time for the WebSphere Application Server to shut down completely before probing the resource to determine if the request to stop was successful. Depending upon the environment, you may need to adjust the OfflineTimeout attribute for this resource to allow the instance ample time to shut down. Properly tuning this attribute ensures that the cluster does not time out the offline entry point while a WebSphere Application Server is completing a graceful shut down.

After a WebSphere Application Server is placed under cluster control, do not attempt to start or stop the instance without using a cluster interface. Only use the Web Console, Java Console, or command-line interface, to start or stop a managed WebSphere instance.

Service group configuration options

The WebSphere deployment type and strategy determines the number of service groups in a cluster required and the number of WebSphere Application Servers managed within each service group. Although not comprehensive, the following examples depict common scenarios to consider.

Figure 3-1 depicts a service group that manages a Deployment Manager Server.

Other service groups manage WebSphere Servers of the type Node agent and Application Server.

Figure 3-1 Service group that manages a Deployment Manager Server

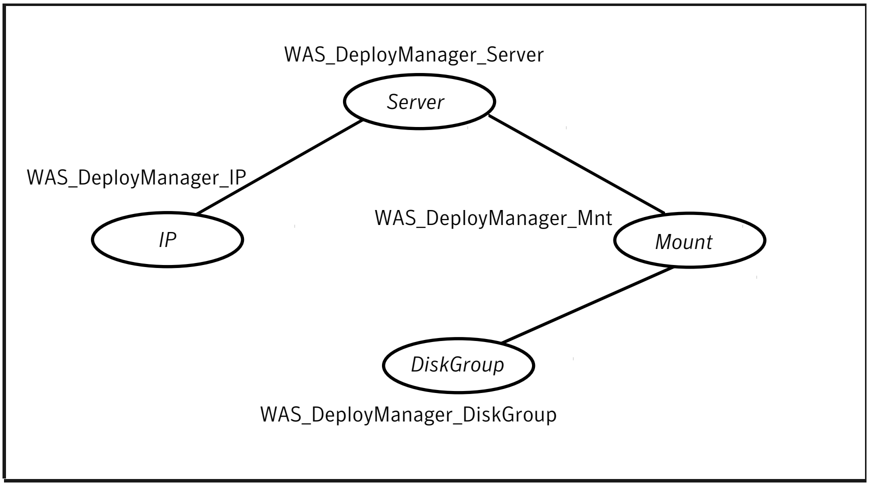


Figure 3-2 depicts a service group that manages a Node Agent Server.

In this configuration, the cluster does not control the Application Servers managed by this Node Agent instance. Thus, the Node Agent Server may fully manage and monitor its managed Application Servers without conflict with the cluster.

Figure 3-2 WebSphere Application Server NodeAgent on all systems

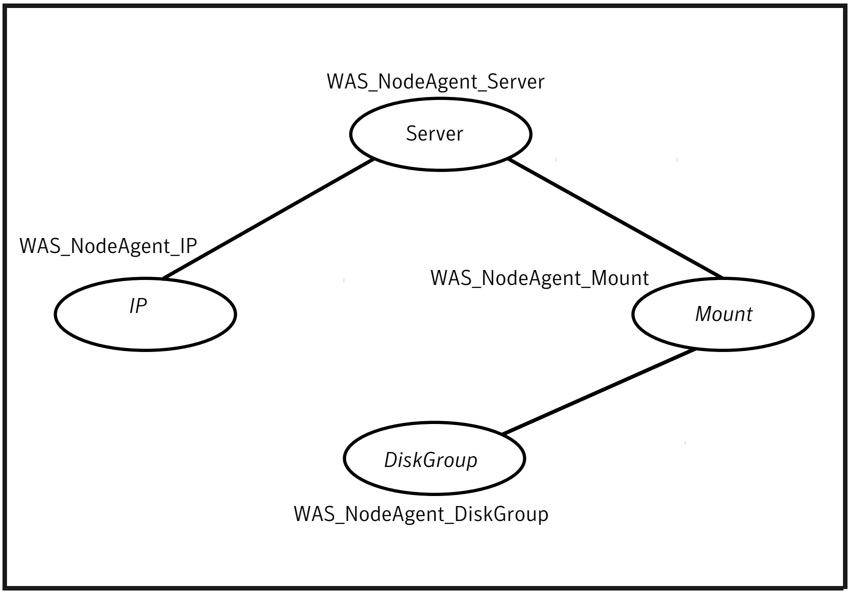
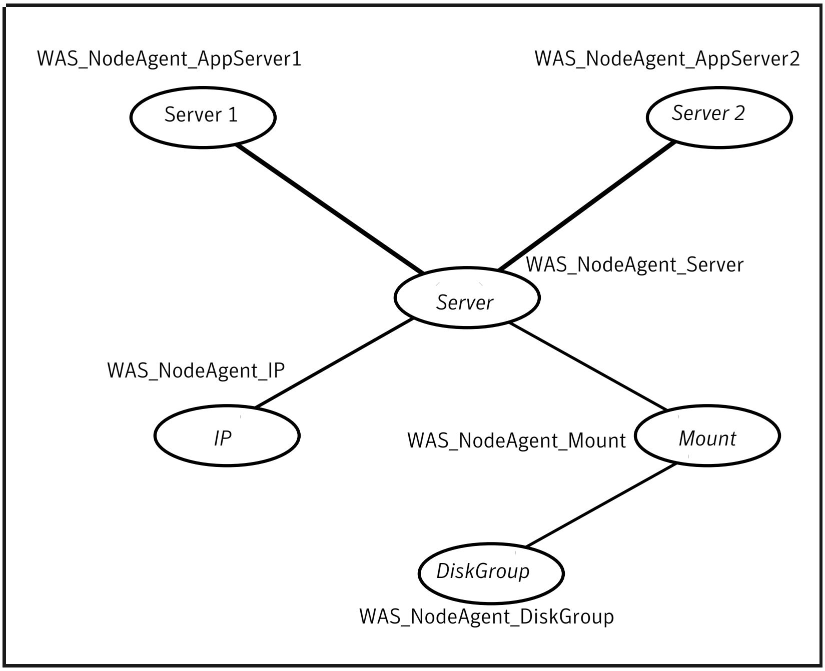


Figure 3-3 depicts a service group that controls a Node Agent Server and its two managed Application Servers.

In this configuration, the cluster controls the Application Servers that are managed by this Node Agent instance. Thus, the Node Agent Server should be configured to not monitor and restart its failed Application Servers, as this would conflict with cluster actions in response to the failure.

Figure 3-3 Service group that controls a Node Agent Server and its two managed Application Servers



Enabling the agent for WebSphere Application Server for IMF

This chapter includes the following topics:

About Intelligent Monitoring Framework

With intelligent monitoring framework (IMF), VCS supports intelligent resource monitoring in addition to the poll-based monitoring. Poll-based monitoring polls the resources periodically whereas intelligent monitoring performs asynchronous monitoring. You can enable or disable the intelligent resource monitoring functionality of the WebSphere Application Server agent.

VCS process and mount-based agents use the Asynchronous Monitoring Framework (AMF) kernel driver that provides asynchronous event notifications to the agents that are enabled for Intelligent Monitoring Framework (IMF).

You can enable the WebSphere Application Server agent for IMF, provided the following software versions are installed:

- Veritas Cluster Server (VCS) 5.1 SP1 or later
- Veritas High Availability agent for WebSphere Application Server version 5.1.9.0 or later

See the *Veritas Cluster Server Administrator's Guide 5.1 Service Pack 1* for more information about:

- IMF notification module functions
- Administering the AMF kernel driver

Benefits of IMF

IMF offers the following benefits:

- Performance
  Enhances performance by reducing the monitoring of each resource at a default of 60 seconds for online resources, and 300 seconds for offline resources. IMF enables the agent to monitor a large number of resources with a minimal effect on performance.
- Faster detection
  Asynchronous notifications would detect a change in the resource state as soon as it happens. Immediate notification enables the agent to take action at the time of the event.

How the agent supports intelligent resource monitoring

When an IMF-enabled agent starts up, the agent initializes the asynchronous monitoring framework (AMF) kernel driver. After the resource is in a steady state, the agent registers with the AMF kernel driver, the details of the resource that are required to monitor the resource. For example, the agent for WebSphere Application Server registers the PIDs of the WebSphere Application Server processes with the IMF notification module. The agent's imf_getnotification function waits for any resource state changes.When the AMF kernel driver module notifies the imf_getnotification function about a resource state change, the agent framework runs the monitor agent function to ascertain the state of that resource. The agent notifies the state change to VCS, which then takes appropriate action.

See the *Veritas Cluster Server Administrator's Guide 5.1 Service Pack 1* for more information.

Agent functions for the IMF functionality

imf_init

This function initializes the WebSphere Application Server agent to interface with the AMF kernel driver, which is the IMF notification module for the agent for WebSphere Application Server. This function runs when the agent starts up.

imf_getnotification

This function gets notifications about resource state changes. This function runs after the agent initializes with the AMF kernel module. This function continuously waits for notification and takes action on the resource upon notification.

imf_register

This function registers or unregisters resource entities with the AMF kernel module. This function runs for each resource after the resource goes into a steady state—online or offline.

Attributes that enable IMF

If the agent for WebSphere Application Server is enabled for IMF, the agent uses the following type-level attributes in addition to the attributes described in WebSphere Application Server agent attributes.

IMF

This resource type-level attribute determines whether the WebSphere Application Server agent must perform intelligent resource monitoring. You can also override the value of this attribute at the resource level.

This attribute includes the following keys:

Mode

Define this attribute to enable or disable intelligent resource monitoring. Valid values are as follows:

- 0—Does not perform intelligent resource monitoring
- 1—Performs intelligent resource monitoring for offline resources and performs poll-based monitoring for online resources
- 2—Performs intelligent resource monitoring for online resources and performs poll-based monitoring for offline resources
- 3—Performs intelligent resource monitoring for both online and for offline resources.

Note: The agent for WebSphere Application Server supports intelligent resource monitoring for online resources only. Hence, Mode should be set to either 0 or 2.

Type and dimension: integer-association

Default: 0

MonitorFreq

This key value specifies the frequency at which the agent invokes the monitor agent function. The value of this key is an integer.

Default: 1

You can set this key to a non-zero value for cases where the agent requires to perform both poll-based and intelligent resource monitoring.

If the value is 0, the agent does not perform poll-based process check monitoring.

After the resource registers with the AMF kernel driver, the agent calls the monitor agent function as follows:

- After every (MonitorFreq x MonitorInterval) number of seconds for online resources
- After every (MonitorFreq x OfflineMonitorInterval) number of seconds for offline resources

RegisterRetryLimit

If you enable intelligent resource monitoring, the agent invokes the imf_register agent function to register the resource with the AMF kernel driver.

The value of the RegisterRetryLimit key determines the number of times the agent must retry registration for a resource. If the agent cannot register the resource within the limit that is specified, then intelligent monitoring is disabled until the resource state changes or the value of the Mode key changes.

Default: 3.

IMFRegList

An ordered list of attributes whose values are registered with the IMF notification module.

Type and dimension: string-vector

Default: No default value

Note: The attribute values can be overriden at the resource level.

Before you enable the agent for IMF

Before you enable the WebSphere Application Server agent for IMF, ensure that the AMF kernel module is loaded and AMF is configured. For details, see the ‘Administering the AMF kernel driver' section of the *Veritas Cluster Server Administrator's Guide 5.1 SP1*. For details about the commands you can use to configure AMF, use the `amfconfig -h` command.

Enabling the agent for IMF

In order to enable the WebSphere Application Server agent for IMF, you must make the following configuration changes to the attributes of the agent:

- AgentFile: Set the AgentFile attribute to Script51Agent
- IMF Mode: Set the IMF Mode attribute to 2
- IMFRegList: Update the IMFRegList attribute

The following sections provide more information on the commands you can use to make these configuration changes, depending on whether VCS is in a running state or not.

Note: If you have upgraded VCS from an earlier version to version 5.1 SP1 or later, and you already have WebSphere Application Server agent version 5.1.9.0 installed, ensure that you run the following commands to create appropriate symbolic links:

```
# cd /opt/VRTSagents/ha/bin/WebSphere
```

```
# ln -s /opt/VRTSamf/imf/imf_getnotification imf_getnotification
```

```
# ln -s /opt/VRTSagents/ha/bin/WebSphere/monitor imf_register
```

If VCS is in a running state

To enable the WebSphere Application Server resource for IMF when VCS is in a running state:

1. Make the VCS configuration writable.

   ```
   # haconf -makerw
   ```

2. Run the following command to update the AgentFile attribute.

   ```
   # hatype -modify WebSphere AgentFile
   /opt/VRTSvcs/bin/Script51Agent
   ```

3. For VCS version 6.0 or later, run the following commands to add the IMF attributes:

   ```
   # haattr -add -static ResourceType IMF -integer -assoc Mode 0
   MonitorFreq 1 RegisterRetryLimit 3
   ```

   ```
   # haattr -add -static ResourceType IMFRegList -string –vector
   ```

   Note: Execute these commands only once after you first enable IMF support for the agent.

4 Run the following command to update the IMF attribute.

```
# hatype -modify WebSphere IMF Mode num MonitorFreq num
RegisterRetryLimit num
```

For example, to enable intelligent monitoring of online resources, with the MonitorFreq key set to 5, and the RegisterRetryLimit key is set to 3, run the following command:

```
# hatype -modify WebSphere IMF Mode 2 MonitorFreq 5
RegisterRetryLimit 3
```

Note: The valid values for the Mode key of the IMF attribute are 0 (disabled) and 2 (online monitoring).

5 Run the following command to update the IMFRegList attribute:

```
# hatype -modify WebSphere IMFRegList WAS_NODE WAS_HOME ServerName
```

6 Save the VCS configuration.

```
# haconf -dump -makero
```

7 If the WebSphere Application Server agent is running, restart the agent.

For information on the commands you can use to restart the agent, see Restarting the agent.

Restarting the agent

To restart the agent:

1 Run the following command to stop the agent forcefully:

```
# haagent -stop WebSphere -force -sys <system>
```

Note: Stopping the agent forcefully eliminates the need to take the resource offline.

2 Run the following command to start the agent:

```
# haagent -start WebSphere -sys <system>.
```

If VCS is not in a running state

To change the WebSphere type definition file when VCS is not in a running state:

1 Update the AgentFile attribute.

```
static str AgentFile = "/opt/VRTSvcs/bin/Script51Agent"
```

2 Update the IMF attribute.

The valid values for the Mode key of the IMF attribute are 0 (disabled) and 2 (online monitoring).

```
static int IMF{} = { Mode=num, MonitorFreq=num,
RegisterRetryLimit=num }
```

For example, to update the IMF attribute such that the Mode key is set to 2, the MonitorFreq key is set to 5, and the RegisterRetryLimit key is set to 3:

```
static int IMF{} = { Mode=2, MonitorFreq=5, RegisterRetryLimit=3
}
```

3 Update the IMFRegList attribute.

```
static str IMFRegList[] = { WAS_NODE WAS_HOME ServerName }
```

Disabling intelligent resource monitoring

To disable intelligent resource monitoring

1 Make the VCS configuration writable.

```
# haconf -makerw
```

2 To disable intelligent resource monitoring for all the resources of a certain type, run the following command:

```
# hatype -modify WebSphere IMF -update Mode 0
```

3 To disable intelligent resource monitoring for a specific resource, run the following command:

```
# hares -override resource_name IMF
```

```
# hares -modify resource_name IMF -update Mode 0
```

4 Save the VCS configuration.

```
# haconf -dump -makero
```

Troubleshooting the configuration for IMF

If you face problems with the IMF configuration or functionality, consider the following:

- Ensure that the following attributes are configured with appropriate values.
  - AgentFile
  - IMF
  - IMFRegList
    If IMFRegList is not configured correctly, the WebSphere Application Server resources that have been registered for IMF get unregistered every time the monitor function is run.
- If you have configured the required attributes to enable the WebSphere Application Server agent for IMF, but the agent is still not IMF-enabled, restart the agent. The imf_init function runs only when the agent starts up, so when you restart the agent, imf_init runs and initializes the WebSphere Application Server agent to interface with the AMF kernel driver.
- You can run the following command to check the value of the MonitorMethod attribute and to verify that a resource is registered for IMF.

  ```
  # hares -value resource MonitorMethod system
  ```

  The MonitorMethod attribute specifies the monitoring method that the agent uses to monitor the resource:
  - Traditional—Poll-based resource monitoring
  - IMF—Intelligent resource monitoring
- You can use the `amfstat` to see a list of registered PIDs for a WebSphere resource.
  Following is a sample output of the `ps -ef` command.

  ```
  # ps -ef| egrep 'server1|nodeagent'

  root     30429     1  0 Nov29 ?        00:00:29 /WAS/AppServer/
  java/bin/java -Declipse.security -Dwas.status.socket=21973 -
  Dosgi.install.area=/WAS/AppServer -Dosgi.configuration.area=/
  WAS/AppServer/profiles/AppSrv01/configuration -Dosgi.framework.
  extensions=com.ibm.cds,com.ibm.ws.eclipse.adaptors -Xshareclasses
  :name=webspherev70_%g,groupAccess,nonFatal -Xscmx50M -Xboot
  classpath/p:/WAS/AppServer/java/jre/lib/ext/ibmorb.jar:/WAS/
  AppServer/java/jre/lib/ext/ibmext.jar -classpath /WAS/AppServer/
  profiles/AppSrv01/properties:/WAS/AppServer/properties:/WAS/
  AppServer/lib/startup.jar:/WAS/AppServer/lib/bootstrap.jar:/WAS/
  ```

```
AppServer/lib/jsf-nls.jar:/WAS/AppServer/lib/lmproxy.jar:/WAS/
AppServer/lib/urlprotocols.jar:/WAS/AppServer/deploytool/itp/
batchboot.jar:/WAS/AppServer/deploytool/itp/batch2.jar:/WAS/
AppServer/java/lib/tools.jar -Dibm.websphere.internalClass
AccessMode=allow -Xms50m -Xmx256m -Xcompressedrefs -Dws.ext.dirs
=/WAS/AppServer/java/lib:/WAS/AppServer/profiles/AppSrv01/classes
:/WAS/AppServer/classes:/WAS/AppServer/lib:/WAS/AppServer/
installedChannels:/WAS/AppServer/lib/ext:/WAS/AppServer/web/
help:/WAS/AppServer/deploytool/itp/plugins/com.ibm.etools.ejbdeploy
/runtime -Dderby.system.home=/WAS/AppServer/derby -Dcom.ibm.itp.
location=/WAS/AppServer/bin -Djava.util.logging.configureByServer=
true -Duser.install.root=/WAS/AppServer/profiles/AppSrv01 -Djavax.
management.builder.initial=com.ibm.ws.management.PlatformMBean
ServerBuilder -Dwas.install.root=/WAS/AppServer -Dpython.cachedir=
/WAS/AppServer/profiles/AppSrv01/temp/cachedir -Djava.util.logging.
manager=com.ibm.ws.bootstrap.WsLogManager -Dserver.root=/WAS/App
Server/profiles/AppSrv01 -Djava.awt.headless=true -Djava.security.
auth.login.config=/WAS/AppServer/profiles/AppSrv01/properties/ws
jaas.conf -Djava.security.policy=/WAS/AppServer/profiles/AppSrv01
/properties/server.policy com.ibm.wsspi.bootstrap.WSPreLauncher -
nosplash -application com.ibm.ws.bootstrap.WSLauncher com.ibm.ws.
runtime.WsServer /WAS/AppServer/profiles/AppSrv01/config Dmgr01
Cell AppSrv01Node nodeagent

root     31996 30429  0 Nov29 ?        00:00:41 /WAS/AppServer/java/
bin/java -Declipse.security -Dwas.status.socket=59734 -Dosgi.install.
area=/WAS/AppServer -Dosgi.configuration.area=/WAS/AppServer/profiles
/AppSrv01/configuration -Djava.awt.headless=true -Dosgi.framework.
extensions=com.ibm.cds,com.ibm.ws.eclipse.adaptors -Xshareclasses:
name=webspherev70_%g,groupAccess,nonFatal -Xscmx50M -Xbootclasspath
/p:/WAS/AppServer/java/jre/lib/ext/ibmorb.jar:/WAS/AppServer/java
/jre/lib/ext/ibmext.jar -classpath /WAS/AppServer/profiles/AppSrv01/
properties:/WAS/AppServer/properties:/WAS/AppServer/lib/startup.jar:/
WAS/AppServer/lib/bootstrap.jar:/WAS/AppServer/lib/jsf-nls.jar:/WAS/
AppServer/lib/lmproxy.jar:/WAS/AppServer/lib/urlprotocols.jar:/WAS/
AppServer/deploytool/itp/batchboot.jar:/WAS/AppServer/deploytool/itp
/batch2.jar:/WAS/AppServer/java/lib/tools.jar -Dibm.websphere.
internalClassAccessMode=allow -Xms50m -Xmx256m -Xcompressedrefs
-Dws.ext.dirs=/WAS/AppServer/java/lib:/WAS/AppServer/profiles/
AppSrv01/classes:/WAS/AppServer/classes:/WAS/AppServer/lib:/WAS/
AppServer/installedChannels:/WAS/AppServer/lib/ext:/WAS/AppServer/
web/help:/WAS/AppServer/deploytool/itp/plugins/com.ibm.etools.
ejbdeploy/runtime -Dderby.system.home=/WAS/AppServer/derby -Dcom.
```

```
ibm.itp.location=/WAS/AppServer/bin -Djava.util.logging.configure
ByServer=true -Duser.install.root=/WAS/AppServer/profiles/AppSrv01
-Djavax.management.builder.initial=com.ibm.ws.management.Platform
MBeanServerBuilder -Dwas.install.root=/WAS/AppServer -Dpython.
cachedir=/WAS/AppServer/profiles/AppSrv01/temp/cachedir -Djava.
util.logging.manager=com.ibm.ws.bootstrap.WsLogManager -Dserver.
root=/WAS/AppServer/profiles/AppSrv01 -Dcom.ibm.security.jgss.
debug=off -Dcom.ibm.security.krb5.Krb5Debug=off -Djava.security.
auth.login.config=/WAS/AppServer/profiles/AppSrv01/properties/
wsjaas.conf -Djava.security.policy=/WAS/AppServer/profiles/
AppSrv01/properties/server.policy com.ibm.wsspi.bootstrap.WS
PreLauncher -nosplash -application com.ibm.ws.bootstrap.WSLauncher
com.ibm.ws.runtime.WsServer /WAS/AppServer/profiles/AppSrv01/config
Dmgr01Cell AppSrv01Node server1root     32215 23059  0 18:07 pts/1
00:00:00 grep WAS
```

The `amfstat` command shows the Queue Manager PIDs monitored by the WebSphere Application Server agent.

```
# amfstat
AMF Status Report

Registered Reapers (2):
=======================
 RID    PID    EVENT    REAPER
 11   28232    2    0    WebSphere

Process ONLINE Monitors (2):
============================
 RID    R_RID    PID    GROUP
 12    11    31996    was_server
 13    11    30429    was_nodeagent
```

- Run the following command to set the ResLogLevel attribute to TRACE. When you set ResLogLevel to TRACE, the agent logs messages in the WebSphere_A.log file.

  ```
  # hares -modify ResourceName ResLogLevel TRACE
  ```

- Run the following command to view the content of the AMF in-memory trace buffer.

  ```
  # amfconfig -p dbglog
  ```

For more troubleshooting information, see the following:

- See "*Troubleshooting the agent for WebSphere Application Server*" on page 55.
- For information about asynchronous monitoring, attributes that enable asynchronous monitoring, and instructions on using some AMF CLI: *Veritas Cluster Server Administrator's Guide 5.1 Service Pack 1*
- *Veritas Cluster Server Release Notes 5.1 Service Pack 1*

Known issues

This release of the agent for WebSphere Application Server has the following known issues:

Problem

An error message might appear when you run the `hares -offline` command to take a resource offline.

Description

When a resource is taken offline, it is unregistered from the AMF module. However, the imf_register function attempts to unregister the resource again.

Workaround

It is safe to ignore this error message.

Chapter 5

Configuring the service groups for WebSphere Application Server

This chapter includes the following topics:

Before configuring the service groups for WebSphere Application Server

Before you configure the WebSphere Application Server service group, you must:

- Verify that VCS is installed and configured on all nodes in the cluster where you will configure the service group.
  Refer to the *Veritas Cluster Server Installation Guide* for more information.
- Verify that the Veritas agent for WebSphere Application Server is installed on all nodes in the cluster.
  See "Installing the agent in a VCS environment" on page 20.

Configuring service groups for WebSphere Application Server

While various methods and procedures can be used to install and cluster a WebSphere Application Server, Symantec recommends the following general process:

Allocating shared disk resource for the WebSphere node

A WebSphere node is a logical group of WebSphere Application Servers that are located on the same physical machine. This machine is also called a host. Multiple WebSphere nodes can exist on a single node.

Symantec recommends installing each WebSphere node to be clustered on a separate, dedicated shared disk resource (e.g. LUN). Work with the appropriate administrative group in your organization to obtain a shared disk resource for the WebSphere node.

Creating a Veritas disk group, volume, and file system

Create a Veritas disk group, volume, and file system on the shared disk resource allocated for the WebSphere node.

Although not recommended, WebSphere Application Servers can be clustered without using Veritas Volume Manager or Veritas File System. But the tight integration between the cluster, Volume Manager, and File System ensures a more comprehensive and resilient high availability solution for your WebSphere Application Server.

Obtaining dedicated virtual IP addresses and host names

Obtain dedicated virtual IP addresses and host names required to support the WebSphere node IP network configuration.

Several configurations are possible. For example, a Node agent, which is an administrative process that manages all servers running on a WebSphere node, can share one IP address and host name with all of its managed servers. Alternatively, the Node agent and each of its managed servers could be assigned its own IP address and host name.

No matter which configuration you deploy, these network addresses and host names will be used exclusively by this WebSphere node, regardless of which system in the cluster is running it.

Obtaining a dedicated user account if needed

If the WebSphere Application Server will not run using the root account, obtain a dedicated UNIX account for the WebSphere Application Server. Refer to the description of attribute User for important instructions and requirements to create the account.

Creating service group and supporting resources

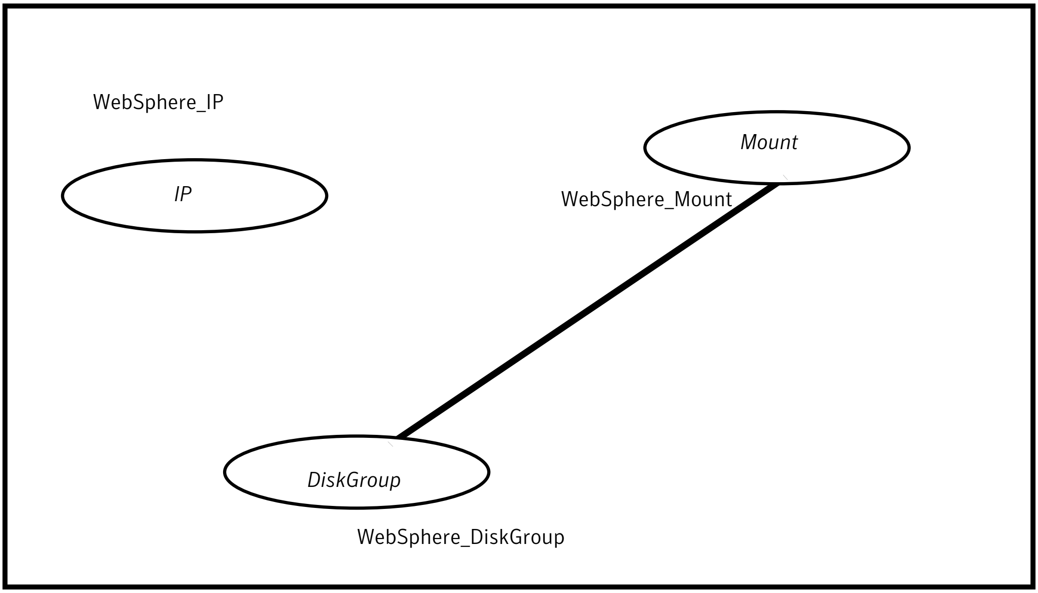
First create a Service Group on a cluster to contain the resources supporting the WebSphere node.

Then create the appropriate cluster resources and links to place the previously created shared disk and networking objects under cluster control.

Test the service group configuration by placing it online. Your service group should appear similar to the following figure.

Figure 5-1 shows a typical service group.

Figure 5-1 Typical service group



Installing the WebSphere software

With the disk and network resources now available and online in the cluster, you are ready to install the WebSphere software.

Follow the instructions in the WebSphere product documentation and install the WebSphere Application Server software. Be sure to instruct the installation program to install the software on the shared disk file system previously established for this WebSphere node.

A well-designed directory structure for your WebSphere Application Server instances will simplify the cluster configuration and create a storage environment that is more intuitive and easier to manage. Assuming that all WebSphere Application Server instances will be clustered and installed on shared disk, Symantec recommends a directory structure similar to the following:

Directory	Purpose
/WAS	Top level directory under which all WebSphere nodes are installed.
/WAS/cell1	Subdirectory under which all WebSphere nodes assigned to cell1 are installed.
/WAS/cell1/depmgr	Subdirectory is the mount point for the shared disk resource dedicated to the Deployment Manager instance supporting cell1.
/WAS/cell1/node1	Subdirectory is the mount point for the shared disk resource dedicated to the WebSphere node named *node1*, which belongs to cell1. The WebSphere software supporting this node agent and its managed Application Servers is installed in this directory.
/WAS/cell1/node2	Subdirectory is the mount point for the shared disk resource dedicated to the WebSphere node named node2, which belongs to cell1. The WebSphere software supporting this node agent and its managed Application Servers is installed in this directory.
/WAS/cell1/node3	Subdirectory is the mount point for the shared disk resource dedicated to the WebSphere node named node3, which belongs to cell1. The WebSphere software supporting this node agent and its managed Application Servers is installed in this directory.

Continue with the same naming pattern for all remaining cells and WebSphere Application Servers.

During the installation, be sure to set the node's Host Name to the dedicated virtual IP host name previously allocated to this node.

Finally, be sure to configure the server's port numbers to avoid conflicts with the port numbers of other WebSphere Application Servers that may be running simultaneously on the same system. Configuring the port numbers is especially important in a cluster environment where WebSphere nodes can be easily moved around the systems in the cluster in almost any combination.

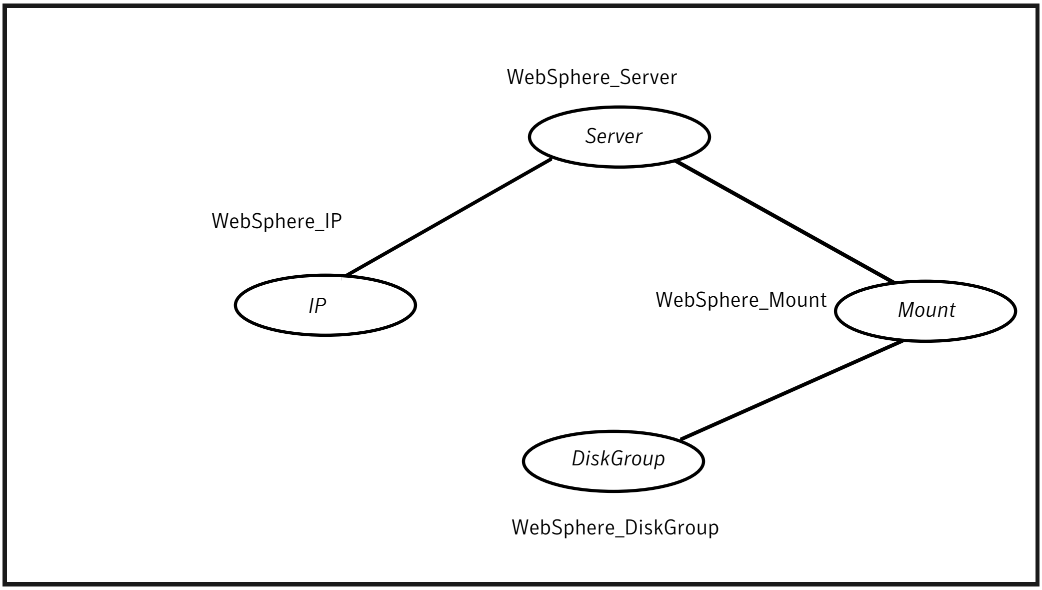
Placing the WebSphere Application Server under cluster control

After the WebSphere Application Server installation is complete, create a cluster resource using the agent for WebSphere to place the server under cluster control.

Your service group should now appear similar to the following figure.

Figure 5-2 shows a typical service group.

Figure 5-2 Typical service group



Warning: After a WebSphere Application Server is placed under cluster control, do not attempt to start or stop the instance without using a cluster interface. Only use the Web Console, Java Console, or command-line interface to start or stop a managed WebSphere instance.

Chapter 6

Troubleshooting the agent for WebSphere Application Server

This chapter includes the following topics:

Using the correct software and operating system versions

Ensure that no issues arise due to incorrect software and operating system versions.

For information on the software versions that the agent for WebSphere Application Server supports, see the Symantec Operations Readiness Tools (SORT) site:https://sort.symantec.com/agents.

Meeting prerequisites

Before installing the agent for WebSphere Application Server, double check that you meet the prerequisites.

For example, you must install the ACC library on VCS before installing the agent for WebSphere Application Server.

Install all the JBoss Application Server components including JBossWS-CXF.

See “Before you install the Veritas agent for WebSphere Application Server” on page 17.

Configuring WebSphere Application Server resources

Before using WebSphere Application Server resources, ensure that you configure the resources properly. For a list of attributes used to configure all WebSphere Application Server resources, refer to the agent attributes.

Starting the WebSphere Application Server instance outside a cluster

If you face problems while working with a resource, you must disable the resource within the cluster framework. A disabled resource is not under the control of the cluster framework, and so you can test the WebSphere Application Server instance independent of the cluster framework. Refer to the cluster documentation for information about disabling a resource.

You can then restart the WebSphere Application Server instance outside the cluster framework.

Note: Use the same parameters that the resource attributes define within the cluster framework while restarting the resource outside the cluster framework.

A sample procedure to start a WebSphere instance outside the cluster framework, is illustrated as follows.

To start a WebSphere Deployment Manager outside the cluster framework

1. Using the user name specified in the User attribute, log into the host on which the WebSphere Deployment Manager application is to run.

2. Use the values specified in the agent attributes to start the WebSphere Deployment Manager.

   For example, assume that the WebSphere Deployment Manager environment is set as follows:

Attribute	Value
ServerType	DeploymentManager
ServerName	dmgr
WAS_NODE	was60c1dmsol
WAS_HOME	/ibm/was/v60/cell1/depmgr
ServerProfile	Dmgr01Specify this attribute for WebSphere version 6.0 and later

3. Go to specified directory.

5.x	/ibm/was/v60/cell1/depmgr/bin
6.x	/ibm/was/v60/cell1/depmgr/profiles/Dmgr0/bin

4. Using the startManager.sh script, start the Deployment Manager.

5.x	`/ibm/was/v60/cell1/depmgr/bin/startManager.sh`
6.x	`/ibm/was/v60/cell1/depmgr/profiles/Dmgr01/bin/startManager.sh`

5. Ensure that the Deployment Manager Server starts successfully.

   If the Deployment Manager works properly outside the cluster framework, you can attempt to implement the server within the framework.

To start a WebSphere node agent outside the cluster framework

1. Using the user name specified in the User attribute, log into the host on which the WebSphere Node agent application is to run.
2. Use the values specified in the agent attributes to start the WebSphere Node agent.

   For example, assume that the WebSphere Node agent environment is set as follows:

Attribute	Value
ServerType	NodeAgent
ServerName	nodeagent
WAS_NODE	was60c1n1sol
WAS_HOME	/ibm/was/v60/cell1/node1
ServerProfile	Default

3. Go to specified directory.

WebSphere version	Directory
5.x	/ibm/was/v60/cell1/node1/bin
6.x	/ibm/was/v60/cell1/node1/profiles/default/bin

4. Using the startNode.sh script, start the Node Agent:

5.x	`/ibm/was/v60/cell1/node1/bin/startNode.sh`
6.x	`/ibm/was/v60/cell1/node1/profiles/default/bin/startNode.sh`

5. Ensure that the Node Agent starts successfully.

   If the Node Agent works properly outside the cluster framework, you can attempt to implement the server within the framework.

Reviewing error log files

If you face problems while using WebSphere Application Server or the agent for WebSphere Application Server, use the log files described in this section to investigate the problems.

Reviewing cluster log files

In case of problems while using the agent for WebSphere Application Server, you can also access the engine log file for more information about a particular resource.

The VCS engine log file is at `/var/VRTSvcs/log/engine_A.log`.

Reviewing agent log files

In case of problems while using the agent for WebSphere, you can access the agent log files for more information. The agent saves output of every entry point process in the temporary folder of the resource system. If the temporary folder is /tmp, the log files are saved using the following naming format:

```
/tmp/.Resource_Name.Entry_Point.Process_ID
```

For example, for a resource WAS50DeployMgr_dmgr:

```
/tmp/.WAS50DeployMgr_dmgr.online.Process_ID
/tmp/.WAS50DeployMgr_dmgr.offline.Process_ID
/tmp/.WAS50DeployMgr_dmgr.clean.Process_ID
/tmp/.WAS50DeployMgr_dmgr.monitor.Process_ID
```

If a resource, WAS50DeployMgr_dmgr is unable to bring a WebSphere Node Manager online, you can access the /tmp/.WAS50DeployMgr_dmgr.online.*Process_ID* for more information so that you can diagnose the problem.

Note: These files are overwritten each time you execute the corresponding agent function process. In case you want to save the information, make a copy of the files at another location.

Using trace level logging

The ResLogLevel attribute controls the level of logging that is written in a cluster log file for each WebSphere Application Server resource. You can set this attribute to TRACE, which enables very detailed and verbose logging.

If you set ResLogLevel to TRACE, a very high volume of messages are produced. Symantec recommends that you localize the ResLogLevel attribute for a particular resource.

To localize ResLogLevel attribute for a resource

1. Identify the resource for which you want to enable detailed logging.
2. Localize the ResLogLevel attribute for the identified resource:

   ```
   # hares -local Resource_Name ResLogLevel
   ```

3. Set the ResLogLevel attribute to TRACE for the identified resource:

   ```
   # hares -modify Resource_Name ResLogLevel TRACE -sys SysA
   ```

4. Note the time before you begin to operate the identified resource.
5. Test the identified resource. The function reproduces the problem that you are attempting to diagnose.
6. Note the time when the problem is reproduced.
7. Set the ResLogLevel attribute back to INFO for the identified resource:

   ```
   # hares -modify Resource_Name ResLogLevel INFO -sys SysA
   ```

8. Review the contents of the log file. Use the time noted in Step 4 and Step 6 to diagnose the problem.

   You can also contact Symantec support for more help.

Defining additional environment variables for a WebSphere Application Server instance

The WebSphere Application Server uses the `setupCmdLine.sh` file, which defines the required environment needed for WebSphere Application Server. The file is sourced in all the start, stop, and monitoring scripts used by the WebSphere Application Server agent.

```
bash-3.00# cat
/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/bin/startServer.sh

#!/bin/sh

binDir=`dirname ${0}`

. ${binDir}/setupCmdLine.sh
```

```
${WAS_HOME}/bin/startServer.sh "$@"
```

By design, the WebSphere Application Server agent does not source the user's proflile. If you need to export any additional environment variables, create an environment file with the required variables and source it in following scripts:

```
startManager.sh, startNode.sh, startServer.sh, stopManager.sh,
stopNode.sh, stopServer.sh, serverStatus.sh
```

For example, append a line in `startServer.sh` in the following manner:

```
# Call User Environement here.

. <Path to env file>/setUserEnv.sh
```

Sample Configurations

This appendix includes the following topics:

About sample configurations for the agent for WebSphere Application Server

The sample configuration graphically depicts the resource types, resources, and resource dependencies within the service group. Review these dependencies carefully before configuring the agent for WebSphere Application Server. For more information about these resource types, see the *Veritas Cluster Server Bundled Agents Reference Guide*.

Sample agent type definition for WebSphere Application Server

VCS 4.x

After importing the agent types into the cluster, if you save the configuration on your system disk using the `haconf -dump` command, you can find the WebSphereTypes.cf file in the /etc/VRTSvcs/conf/config cluster configuration directory.

An excerpt from this file follows.

```
type WebSphere (
   static str ArgList [] = { ResLogLevel, State, IState, ServerName,
      WAS_NODE, WAS_HOME ,User, ServerProfile, ServerType,
      StartOptions, StopOptions, MonitorProgram, SecondLevelMonitor}
   str  ResLogLevel = INFO
   str  ServerName
   str  WAS_NODE
   str  WAS_HOME
   str  User
   str  ServerProfile
   str  ServerType
   str  StartOptions
   str  StopOptions
   str  MonitorProgram
   int  SecondLevelMonitor
   )
```

VCS 5.x

After importing the agent types into the cluster, if you save the configuration on your system disk using the `haconf -dump` command, you can find the WebSphereTypes.cf file in the /etc/VRTSagents/ha/conf/config cluster configuration directory.

An excerpt from this file follows.

```
type WebSphere (
    static str AgentFile = "/opt/VRTSvcs/bin/Script50Agent"
    static str AgentDirectory = "/opt/VRTSagents/ha/bin/WebSphere"
   static str ArgList [] = { ResLogLevel, State, IState, ServerName,
      WAS_NODE, WAS_HOME ,User, ServerProfile, ServerType,
      StartOptions, StopOptions, MonitorProgram, SecondLevelMonitor}
   str  ResLogLevel = INFO
   str  ServerName
   str  WAS_NODE
   str  WAS_HOME
   str  User
   str  ServerProfile
   str  ServerType
   str  StartOptions
   str  StopOptions
   str  MonitorProgram
   int  SecondLevelMonitor
   )
```

Sample configuration in a VCS environment

The following is an excerpt from a VCS configuration file (main.cf) that defines a Network Deployment of WebSphere Application Servers and two independent Application Servers (Application Servers that are not part of a Network Deployment).

This configuration demonstrates that you can combine Network Deployment WebSphere Cells with independent WebSphere Application Servers. In the example, there is one WebSphere Cell consisting of one Deployment Manager named `dmgr`. The WebSphere Cell contains two Node Managers, both named `nodeagent`.

Review the information to configure a service group that manages one independent, stand-alone Application Server.

See "Service group configuration options" on page 33.

```
group WAS51Cell1DM (
        SystemList = { sysa = 0, sysb = 1, sysc = 2 }
        )
        DiskGroup WAS51Cell1DM_dg (
                DiskGroup = was51c1dm
                )

        IP WAS51Cell1DM_ip (
                Device = hme0
                Address = "10.136.228.11"
                NetMask = "255.255.248.0"
                )

        Mount WAS51Cell1DM_mnt (
                MountPoint = "/WAS51/cell1/depmgr"
                BlockDevice = "/dev/vx/dsk/was51c1dm/was"
                FSType = vxfs
                FsckOpt = "-y"
                )

        WebSphere WAS51Cell1DM_was (
                ServerName = dmgr
                WAS_NODE = was51c1dm
                WAS_HOME = "/WAS51/cell1/depmgr"
                User = root
                ServerType = DeploymentManager
                )

        WAS51Cell1DM_was requires WAS51Cell1DM_ip
        WAS51Cell1DM_was requires WAS51Cell1DM_mnt
        WAS51Cell1DM_mnt requires WAS51Cell1DM_dg


group WAS51Cell1Node1 (
        SystemList = { sysb = 0, sysa = 1, sysc = 2 }
        )
        DiskGroup WAS51Cell1Node1_dg (
                DiskGroup = was51c1n1
                )

        IP WAS51Cell1Node1_ip (
                Device = hme0
                Address = "10.136.228.18"
                NetMask = "255.255.248.0"
                )

        Mount WAS51Cell1Node1_mnt (
                MountPoint = "/WAS51/cell1/node1"
                BlockDevice = "/dev/vx/dsk/was51c1n1/was"
                FSType = vxfs
                FsckOpt = "-y"
                )

        WebSphere WAS51Cell1Node1_NA_was (
                ServerName = nodeagent
                WAS_NODE = was51c1n1
                WAS_HOME = "/WAS51/cell1/node1"
                User = root
                ServerType = NodeAgent
                )

        WebSphere WAS51Cell1Node1_AS1_was (
                ServerName = server1
                WAS_NODE = was51c1n1
                WAS_HOME = "/WAS51/cell1/node1"
                User = root
                ServerType = ApplicationServer
                )
        WebSphere WAS51Cell1Node1_AS2_was (
                ServerName = server2
                WAS_NODE = was51c1n1
                WAS_HOME = "/WAS51/cell1/node1"
                User = root
                ServerType = ApplicationServer
                )

        WAS51Cell1Node1_AS2_was requires WAS51Cell1Node1_NA_was
        WAS51Cell1Node1_mnt requires WAS51Cell1Node1_dg
        WAS51Cell1Node1_NA_was requires WAS51Cell1Node1_ip
        WAS51Cell1Node1_NA_was requires WAS51Cell1Node1_mnt
        WAS51Cell1Node1_AS1_was requires WAS51Cell1Node1_NA_was


group WAS51Cell1Node2 (
        SystemList = { sysa = 0, sysa = 1, sysb = 2 }
        )

        DiskGroup WAS51Cell1Node2_dg (
                DiskGroup = was51c1n2
                )

        IP WAS51Cell1Node2_ip (
                Device = eri0
                Address = "10.136.228.19"
                NetMask = "255.255.248.0"
                )

        Mount WAS51Cell1Node2_mnt (
                MountPoint = "/WAS51/cell1/node2"
                BlockDevice = "/dev/vx/dsk/was51c1n2/was"
                FSType = vxfs
                FsckOpt = "-y"
                )

        WebSphere WAS51Cell1Node2_NA_was (
                ServerName = nodeagent
                WAS_NODE = was51c1n2
                WAS_HOME = "/WAS51/cell1/node2"
                User = root
                ServerType = NodeAgent
                )
        WebSphere WAS51Cell1Node2_AS1_was (
                ServerName = server1
                WAS_NODE = was51c1n2
                WAS_HOME = "/WAS51/cell1/node2"
                User = root
                ServerType = ApplicationServer
                )

        WAS51Cell1Node2_mnt requires WAS51Cell1Node2_dg
        WAS51Cell1Node2_NA_was requires WAS51Cell1Node2_ip
        WAS51Cell1Node2_NA_was requires WAS51Cell1Node2_mnt
        WAS51Cell1Node2_AS1_was requires WAS51Cell1Node2_NA_was


group WAS51AppSrvr1 (
        SystemList = { sysa = 0, sysb = 1, sysc = 2 }
        )

        DiskGroup WAS51AppSrvr1_dg (
                DiskGroup = was51appsrvr1
                )

        IP WAS51AppSrvr1_ip (
                Device = hme0
                Address = "10.136.228.21"
                NetMask = "255.255.248.0"
                )

        Mount WAS51AppSrvr1_mnt (
                MountPoint = "/WAS51/srvr1"
                BlockDevice = "/dev/vx/dsk/was51appsrvr1/was"
                FSType = vxfs
                FsckOpt = "-y"
                )

        WebSphere WAS51AppSrvr1_AppSrvr_was (
                ServerName = server1
                WAS_NODE = was51srv1
                WAS_HOME = "/WAS51/srvr1"
                User = root
                ServerType = ApplicationServer
                )
        WAS51AppSrvr1_mnt requires WAS51AppSrvr1_dg
        WAS51AppSrvr1_AppSrvr_was requires WAS51AppSrvr1_ip
        WAS51AppSrvr1_AppSrvr_was requires WAS51AppSrvr1_mnt


group WAS51AppSrvr2 (
        SystemList = { sysa = 0, sysb = 1, sysc = 2 }
        )

        DiskGroup WAS51AppSrvr2_dg (
                DiskGroup = was51appsrvr2
                )

        IP WAS51AppSrvr2_ip (
                Device = hme0
                Address = "10.136.228.22"
                NetMask = "255.255.248.0"
                )

        Mount WAS51AppSrvr2_mnt (
                MountPoint = "/WAS51/srvr2"
                BlockDevice = "/dev/vx/dsk/was51appsrvr2/was"
                FSType = vxfs
                FsckOpt = "-y"
                )

        WebSphere WAS51AppSrvr2_AppSrvr_was (
                ServerName = server1
                WAS_NODE = was51srv1
                WAS_HOME = "/WAS51/srvr2"
                User = root
                ServerType = ApplicationServer
                )

        WAS51AppSrvr2_mnt requires WAS51AppSrvr2_dg
        WAS51AppSrvr2_AppSrvr_was requires WAS51AppSrvr2_ip
        WAS51AppSrvr2_AppSrvr_was requires WAS51AppSrvr2_mnt
```

Appendix B

Changes introduced in previous releases

This appendix includes the following topics:

- Changes introduced in previous releases

Changes introduced in previous releases

The enhancements in the previous releases of Veritas agent for WebSphere are as follows:

- Added Intelligent Monitoring Framework (IMF) capability and support for intelligent resource monitoring.
- The SecondLevelMonitor attribute of the Veritas agent for WebSphere Application Server is deprecated if both the following software versions are installed:
  - Veritas Cluster Server (VCS) 5.1 SP1or later
  - Veritas agent for WebSphere Application Server version 5.1.9.0

  In such a case, instead of the SecondLevelMonitor attribute, you must use the resource type level attribute—LevelTwoMonitorFreq—to specify the frequency of in-depth monitoring.
- Added support for AIX 7.1 and Red Hat Enterprise Linux 6.0.
- Fixed issue that arose due to incorrect timeout value. The WebSphere agent was using incorrect timeout values in certain cases to set the alarm for execution of the commands. Now the agent is using the correct timeout value for the entry point operations.
- Added support for IPv6.

- Added support for WebSphere Application Server 7.0.
- Added support for AIX 6.1 on pSeries.
- Added support for VCS 5.1 on AIX, Linux, and Solaris.
- Added support for SUSE Linux Enterprise Server 9.
- Added support for VCS One 2.0 and 5.0.
- Added support for Solaris x86 for VCS 4.1 and 5.0.
- Added support for Internationalization (i18n).
- Added support for VCS version 5.0.
- Added support for ACC library 5.1 that is compliant with VCS 4.x.
- Added support for zones on Solaris 10 in a VCS environment.
- Renamed agent type WebSphere5 to agent type WebSphere.

Index