Symantec™ High Availability Agent for PostgreSQL Installation and Configuration Guide

AIX, HP-UX, Linux, Solaris SPARC

5.1

June 2014

Symantec High Availability Agent for PostgreSQL Installation and Configuration Guide

Agent Version: 5.1.5.0

Document version: 5.1.5.0.1

Legal Notice

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 website 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 website.

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

For information regarding the latest HOWTO articles, documentation updates, or to ask a question regarding product documentation, visit the Storage and Clustering Documentation forum on Symantec Connect.

https://www-secure.symantec.com/connect/storage-management/forums/storage-and-clustering-documentation

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 Symantec High Availability Agent for PostgreSQL

This chapter includes the following topics:

About the Symantec High Availability agent for PostgreSQL

Symantec 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 Symantec High Availability agent for PostgreSQL manages and provides high availability for PostgreSQL servers and EnterpriseDB Postgres Plus Advanced Servers in a clustered environment.

The agent can bring a specific PostgreSQL server instance online and monitor the state of the PostgreSQL server. The agent can also detect failures and can shut down the instance in case of a failure.

What's new in this agent

The enhancements in this release of the agent are:

- Symantec has introduced an optional attribute —StopOpts— that gives options for shutting down the PostgreSQL database server.

Supported software

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

Support matrix for IMF and in-depth monitoring

Depending on your version of Symantec Cluster Server (VCS) and the PostgreSQL agent, the following features and functionality are supported.

Table 1-1 IMF and detail monitoring support matrix

VCS and agent version	IMF capability	SecondLevelMonitor attribute	LevelTwoMonitorFreq attribute
VCS 5.1 SP1 or later with PostgreSQL agent 5.1.1.0 or later	Yes	No	Yes
VCS 5.1 SP1 with PostgreSQL agent 5.1.0.0	No	Yes	No
VCS 5.1 or earlier with PostgreSQL agent 5.1.1.0 or later	No	Yes	No
VCS 5.1 or earlier with PostgreSQL agent 5.1.0.0	No	Yes	No

Features of the agent

The following are the features of the Symantec High Availability agent for PostgreSQL:

- Support for validation of attributes that are based on agent functions.
  The agent can validate attributes in each agent function before the actual data processing starts.
- Support for First Failure Data Capture (FFDC)
  In case of a fault, the agent generates a huge volume of the debug logs that enable troubleshooting of the fault.
- Support for Fast First Level Monitor (FFLM)
  The agent maintains PID files based on search patterns to expedite the monitoring process.
- Support for external user-supplied monitor utilities
  The agent enables user-specified monitor utilities to be plugged in, in addition to the built-in monitoring logic. This enables administrators to completely customize the monitoring of the application.
- Support for intelligent resource monitoring and poll-based monitoring.
  The agent supports the VCS Intelligent Monitoring Framework (IMF) feature. IMF allows the agent to register the resources to be monitored with the IMF notification module so as to receive immediate notification of resource state changes without having to periodically poll the resources. See "*Enabling the agent for PostgreSQL to support IMF*" on page 39.
- Delayed agent function
  The agent manages the first monitor after online for slow initializing applications.

How the agent supports intelligent resource monitoring

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.

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 PostgreSQL registers the PIDs of the PostgreSQL processes with the AMF kernel driver. 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 *Symantec Cluster Server Administrator's Guide* for more information.

PostgreSQL agent functions

The operations or functions that the Symantec High Availability agent for PostgreSQL can perform are as follows:

Online

The online function performs the following tasks:

- Verifies that the required attributes are set correctly.
- Verifies that the PostgreSQL server instance is not already online. If the instance is online, the online operation exits immediately.
- If any PostgreSQL processes remain, the operation kills these processes using the user name associated with the specific resource.
- Attempts to start the PostgreSQL server instance with the command:

  ```
  $ BaseDir/pg_ctl start -w -D DataDir -o "-p Port -h
  HostName" StartOpts
  ```

  The command always gets executed in the context of a PostgreSQL user, who has the privileges to start and stop the postgres (postmaster) process.
- Checks if the server has started up completely.
- Gives the control back to HAD.

Offline

The offline function performs the following tasks:

- Verifies that the required attributes are set correctly.
- Verifies that the PostgreSQL server instance is not offline.
- If the instance is already offline, the operation verifies if any processes belonging to this PostgreSQL resource exist.
- Attempts to stop the PostgreSQL server instance with the command:

```
$ BaseDir/pg_ctl stop -w -D DataDir StopOpts
```

The `pg_ctl` command uses the option specified in the StopOpts attribute to shut down the PostgreSQL database server. If no option is specified in the StopOpts attribute, the agent stops the database server using the default `-m smart` shutdown option.

The command always gets executed in the context of a PostgreSQL user, who has the privileges to start and stop the postgres (postmaster) process.

- Gives the control back to HAD.

Monitor

The monitor function monitors the states of the PostgreSQL servers on all nodes within the cluster. The operation performs the following tasks:

- The monitor function conducts a first-level check to determine that the PostgreSQL server processes are running on the system in the cluster. If the first-level check does not find these processes running on the node, the check exits immediately and reports the instance as OFFLINE.

  The agent for PostgreSQL also supports Intelligent Monitoring Framework (IMF) in the first-level check. IMF enables intelligent resource monitoring. The agent for PostgreSQL is IMF-aware and uses the asynchronous monitoring framework (AMF) kernel driver for resource state change notifications. See "How the agent supports intelligent resource monitoring" on page 12. 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 41.

- If the SecondLevelMonitor attribute is set to greater than 0, the monitor operation conducts a second level check.

  During Second Level Monitoring, the agent uses the monitor command to verify that the PostgreSQL server is really up.

  ```
  $ BaseDir/pg_ctl status -D DataDir
  ```

  The command is executed in the context of a PostgreSQL user, who has the privilege to monitor the postgres (postmaster) process.

Note: The attribute used to configure the second-level check and its frequency depends on the software versions of VCS and PostgreSQL agent you have installed: For VCS 5.1 SP1 or later with PostgreSQL agent version 5.1.1.0, use the LevelTwoMonitorFreq attribute. For VCS 5.1 or earlier with PostgreSQL agent 5.1.0.0 or earlier, use the SecondLevelMonitor attribute.

- Depending upon the value of the MonitorProgram attribute, the monitor operation can perform a customized check using a user-supplied monitoring utility. Refer to the agent attributes for more details regarding this attribute: See "PostgreSQL agent attributes" on page 30.

Clean

In case of a failure or after an unsuccessful attempt to bring a PostgreSQL server instance online or take a PostgreSQL server instance offline , the clean operation performs the following tasks:

- Attempts to gracefully shut down the PostgreSQL server instance with the command:

  ```
  $ BaseDir/pg_ctl stop -w -D DataDir
  ```

  The command always gets executed in the context of a PostgreSQL user, who has the privileges to start and stop the postgres (postmaster) process.
- The clean operation kills any remaining processes pertaining to this PostgreSQL instance.
- Gives the control back to HAD.

Note: For information about the additional functions of the agent for PostgreSQL when IMF is enabled: See "Agent functions for the IMF functionality" on page 40.

Chapter 2

Installing, upgrading, and removing the agent for PostgreSQL

This chapter includes the following topics:

Before you install the Symantec High Availability agent for PostgreSQL

You must install the Symantec High Availability agent for PostgreSQL on all the systems that will host PostgreSQL service groups.

Ensure that you meet the following prerequisites to install the agent for PostgreSQL.

- Install and configure Symantec Cluster Server.
  For more information on installing and configuring Symantec Cluster Server, refer to the *Symantec Cluster Server Installation Guide*.

- 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 17.
- On Solaris 11, ensure that the pkg:/compatibility/ucb package is installed on the system.

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 tar file or the individual ACCLib tar file from the Symantec Operations Readiness Tools (SORT) site (https://sort.symantec.com/agents).
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/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

4 If you downloaded the individual ACCLib tar file, 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 ACC library IPS package on Oracle Solaris 11 systems

To install the ACC library IPS package on an Oracle Solaris 11 system

1 Copy the VRTSacclib.p5p package from the pkgs directory to the system in the `/tmp/install` directory.

2 Disable the publishers that are not reachable as package install may fail if any of the already added repositories are unreachable.

```
# pkg set-publisher --disable <publisher name>
```

3 Add a file-based repository in the system.

```
# pkg set-publisher -g /tmp/install/VRTSacclib.p5p Symantec
```

4 Install the package.

```
# pkg install --accept VRTSacclib
```

5 Remove the publisher from the system.

```
# pkg unset-publisher Symantec
```

6 Enable the publishers that were disabled earlier.

```
# pkg set-publisher --enable <publisher name>
```

Installing the ACC library package on Solaris brand non-global zones

With Oracle Solaris 11, you must install the ACC library package inside non-global zones. The native non-global zones are called Solaris brand zones.

To install the ACC library package on Solaris brand non-global zones

1. Ensure that the SMF service `svc:/application/pkg/system-repository:default` and `svc:/application/pkg/zones-proxyd:default` are online on the global zone.

   ```
   # svcs svc:/application/pkg/system-repository:default
   ```

   ```
   # svcs svc:/application/pkg/zones-proxyd:default
   ```

2. Log on to the non-global zone as a superuser.
3. Ensure that the SMF service `svc:/application/pkg/zones-proxy-client:default` is online inside non-global zone:

   ```
   # svcs svc:/application/pkg/zones-proxy-client:default
   ```

4. Copy the VRTSacclib.p5p package from the pkgs directory to the non-global zone (for example at `/tmp/install` directory).
5. Disable the publishers that are not reachable, as package install may fail if any of the already added repositories are unreachable.

   ```
   # pkg set-publisher --disable <publisher name>
   ```

6. Add a file-based repository in the non-global zone.

   ```
   # pkg set-publisher -g/tmp/install/VRTSacclib.p5p Symantec
   ```

7. Install the package.

   ```
   # pkg install --accept VRTSacclib
   ```

8. Remove the publisher on the non-global zone.

   ```
   # pkg unset-publisher Symantec
   ```

9. Clear the state of the SMF service, as setting the file-based repository causes the SMF service `svc:/application/pkg/system-repository:default` to go into maintenance state.

   ```
   # svcadm clear svc:/application/pkg/system-repository:default
   ```

10. Enable the publishers that were disabled earlier.

    ```
    # pkg set-publisher --enable <publisher>
    ```

Note: Perform steps 2 through 10 on each non-global zone.

Installing the agent in a VCS environment

Install the agent for PostgreSQL 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/postgresql_agent/`*`vcs_version`*`/`*`version`*`_agent/pkgs`
HP-UX	`cd1/hpux/generic/vcs/application/postgresql_agent/`*`vcs_version`*`/`*`version`*`_agent/pkgs`
Linux	`cd1/linux/generic/vcs/application/postgresql_agent/`*`vcs_version`*`/`*`version`*`_agent/rpms`
Solaris	`cd1/solaris/`*`dist_arch`*`/vcs/application/postgresql_agent/`*`vcs_version`*`/`*`version`*`_agent/pkgs`

   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 VRTSpgsql.rte.bff VRTSpgsql.rte`
HP-UX	`# swinstall -s 'pwd' VRTSpgsql`
Linux	`# rpm -ihv \` `VRTSpgsql-`*`AgentVersion`*`-GA_GENERIC.noarch.rpm`
Solaris	`# pkgadd -d . VRTSpgsql`

6 After installing the agent package, you must import the agent type configuration file.

Installing the agent IPS package on Oracle Solaris 11 systems

To install the agent IPS package on an Oracle Solaris 11 system

1 Copy the VRTSpgsql.p5p package from the pkgs directory to the system in the `/tmp/install` directory.

2 Disable the publishers that are not reachable as package install may fail if any of the already added repositories are unreachable.

```
# pkg set-publisher --disable <publisher name>
```

where the publisher name is obtained using the `pkg publisher` command.

3 Add a file-based repository in the system.

```
# pkg set-publisher -g /tmp/install/VRTSpgsql.p5p Symantec
```

4 Install the package

```
# pkg install --accept VRTSpgsql
```

5 Remove the publisher from the system.

```
# pkg unset-publisher Symantec
```

6 Enable the publishers that were disabled earlier.

```
# pkg set-publisher --enable <publisher name>
```

Installing agent packages on Solaris brand non-global zones

With Oracle Solaris 11, you must install the agent package inside non-global zones. The native non-global zones are called Solaris brand zones.

To install the agent package on Solaris brand non-global zones

1. Ensure that the SMF service`svc:/application/pkg/system-repository:default` and `svc:/application/pkg/zones-proxyd:default` are online on the global zone.

   ```
   # svcs svc:/application/pkg/system-repository:default
   ```

   ```
   # svcs svc:/application/pkg/zones-proxyd:default
   ```

2. Log on to the non-global zone as a superuser.
3. Ensure that the SMF service `svc:/application/pkg/zones-proxy-client:default` is online inside non-global zone:

   ```
   # svcs svc:/application/pkg/zones-proxy-client:default
   ```

4. Copy the VRTSpgsql.p5p package from the pkgs directory to the non-global zone (for example at `/tmp/install` directory).
5. Disable the publishers that are not reachable, as package install may fail if any of the already added repositories are unreachable.

   ```
   # pkg set-publisher --disable <publisher name>
   ```

6. Add a file-based repository in the non-global zone.

   ```
   # pkg set-publisher -g/tmp/install/VRTSpgsql.p5p Symantec
   ```

7. Install the package.

   ```
   # pkg install --accept VRTSpgsql
   ```

8. Remove the publisher on the non-global zone.

   ```
   # pkg unset-publisher Symantec
   ```

9. Clear the state of the SMF service, as setting the file-based repository causes the SMF service `svc:/application/pkg/system-repository:default` to go into maintenance state.

   ```
   # svcadm clear svc:/application/pkg/system-repository:default
   ```

10. Enable the publishers that were disabled earlier.

    ```
    # pkg set-publisher --enable <publisher>
    ```

Note: Perform steps 2 through 10 on each non-global zone.

Installing the agent in a non-global zone on Solaris 11

To install the PostgreSQL agent in a non-global zone on Solaris 11:

- Ensure that the ACClibrary package, VRTSacclib, is installed in the non-global zone.
  To install VRTSacclib in the non-global zone, run the following command from the global zone:
  ```
  # pkgadd -R /zones/zone1/root -d VRTSacclib.pkg
  ```
- To install the agent package in the non-global zone, run the following command from the global zone:
  ```
  # pkgadd -R zone-root/root -d . VRTSpgsql
  ```
  For example: `# pkgadd -R /zones/zone1/root -d . VRTSpgsql`

Note: You can ignore the following messages that might appear:

```
## Executing postinstall script.

ln: cannot create
/opt/VRTSagents/ha/bin/PostgreSQL/imf_getnotification: File exists

ln: cannot create /opt/VRTSagents/ha/bin/PostgreSQL/imf_register:
File exists

or ## Executing postinstall script.

ln: cannot create
/opt/VRTSagents/ha/bin/PostgreSQL/imf_getnotification: No such file
or directory

ln: cannot create /opt/VRTSagents/ha/bin/PostgreSQL/imf_register: No
such file or directory
```

Uninstalling the agent in a VCS environment

You must uninstall the agent for PostgreSQL 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 PostgreSQL resources from the cluster. Use the following command to verify that all resources have been removed:

```
# hares -list Type=PostgreSQL
```

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

```
# hatype -delete PostgreSQL
```

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 PostgreSQL from each node in the cluster.

Execute the following command to uninstall the agent:

AIX	# `installp -u VRTSpgsql.rte`
HP-UX	# `swremove VRTSpgsql`
Linux	# `rpm -e VRTSpgsql`
Solaris	# `pkgrm VRTSpgsql` **Note:** To uninstall the agent IPS package on a Solaris 11 system: `# pkg uninstall VRTSpgsql`

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`
	Note: To uninstall the ACClib IPS package on a Solaris 11 system:
	`# pkg uninstall VRTSacclib`

Upgrading the agent in a VCS environment

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

To upgrade the agent in a VCS environment

1. Persistently freeze the service groups that host the application.

   ```
   # hagrp -freeze GroupName -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 PostgreSQL
   ```

4 Uninstall the agent package from all the nodes. Use the platform's native software management program to remove the agent for PostgreSQL from each node in the cluster.

Execute the following command to uninstall the agent:

AIX	# `installp -u VRTSpgsql.rte`
HP-UX	# `swremove VRTSpgsql`
Linux	# `rpm -e VRTSpgsql`
Solaris	For Solaris 10: # `pkgrm VRTSpgsql` For Solaris 11: # **`pkg uninstall VRTSpgsql`**

5 Install the new agent on all the nodes.

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

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

VCS 5.x	■ AIX ■ HP-UX ■ Linux	`/etc/VRTSagents/ha/conf/PostgreSQL/PostgreSQLTypes.cf`
VCS 5.0	■ Solaris SPARC	`/etc/VRTSagents/ha/conf/PostgreSQL/PostgreSQLTypes50.cf`
VCS 5.1	■ Solaris SPARC	`/etc/VRTSagents/ha/conf/PostgreSQL/PostgreSQLTypes51.cf`

Note: If you upgraded the VCS version to VCS 5.1 SP1 and the PostgreSQL agent version to 5.1.1.0 (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.

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

Note: To note the list of changed attributes, compare the new type definition file with the old type definition file.

8 Start the cluster services.

```
# hastart
```

9 Start the agent on all nodes, if not started.

```
# haagent -start PostgreSQL -sys SystemName
```

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

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

Chapter 3

Configuring the agent for PostgreSQL

This chapter includes the following topics:

About configuring the Symantec High Availability agent for PostgreSQL

After installing the Symantec High Availability agent for PostgreSQL, you must import the agent type configuration file. After importing this file, review the attributes table that describes the resource type and its attributes, and then create and configure PostgreSQL resources.

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

See "About sample configurations for the agent for PostgreSQL" on page 57.

Importing the agent types files in a VCS environment

To use the agent for PostgreSQL, you must import the agent types file into the cluster.

You can import the agent types file using the Symantec 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 Cluster Manager (Java Console) 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 5.x or later	■ AIX ■ HP-UX ■ Linux	`/etc/VRTSagents/ha/conf/PostgreSQL/PostgreSQLTypes.cf`
VCS 5.0	Solaris SPARC	`/etc/VRTSagents/ha/conf/PostgreSQL/PostgreSQLTypes50.cf`
VCS 5.1 or later	Solaris SPARC	`/etc/VRTSagents/ha/conf/PostgreSQL/PostgreSQLTypes51.cf`

4. Click **Import**.
5. Save the VCS configuration.

   The PostgreSQL agent type is now imported to the VCS engine.

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

To import the agent types file using the 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 5.x or later	■ AIX ■ HP-UX ■ Linux	`/etc/VRTSagents/ha/conf/PostgreSQL/PostgreSQLTypes.cf`
VCS 5.0	■ Solaris SPARC	`/etc/VRTSagents/ha/conf/PostgreSQL/PostgreSQLTypes50.cf`

VCS 5.1 or later	■ Solaris SPARC	/etc/VRTSagents/ha/conf/PostgreSQL/PostgreSQLTypes51.cf

4 Create a dummy main.cf file:

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

5 Create the PostgreSQL resource type as follows:

```
# hacf -verify .

# haconf -makerw

# sh main.cmd

# haconf -dump
```

The PostgreSQL agent type is now imported to the VCS engine.

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

PostgreSQL agent attributes

Refer to the required and optional attributes while configuring the agent for PostgreSQL.

Table 3-1 lists the required attributes for the PostgreSQL agent.

Table 3-1 Required attributes

Attribute	Description
ResLogLevel	Specifies the logging detail that the agent performs for the resource. The valid values are as follows: ■ ERROR: Only logs error messages. ■ WARN: Logs error messages and warning messages. ■ INFO: Logs error messages, warning messages, and informational messages ■ TRACE: Logs error messages, warning messages, informational messages, and trace messages. TRACE is very verbose and should be used only during initial configuration or for troubleshooting and diagnostic operations. Default Value: INFO Example: INFO

Table 3-1 Required attributes *(continued)*

Attribute	Description
PostgreSQLUser	The dedicated OS login created while installing the PostgreSQL server. All the database server operations, such as start, stop, and monitor, will be done as this user. This login must be identical on all failover nodes. **Note:** For EnterpriseDB PostgreSQL plus Advanced Server, the default value of this attribute is enterprisedb. Modify the value of this attribute as required. Default Value: postgres Example: postgres
DataDir	The absolute path to the directory storing the database being managed by this instance of the server. Symantec recommends storing this directory on shared storage so that the same copy is available on the failover node. The database directory should be owned by the PostgreSQL user. Default Value: "" Example: /opt/postgres/data
BaseDir	The installation path of the PostgreSQL Database server, where the PostgreSQL executables (pg_ctl, postgres, and so on) reside . Default Value: "" Example: /usr/bin
HostName	Virtual host name for this PostgreSQL Database instance. The database server is started using the virtual host name and port number provided. Default Value: "" Example: web1.veritas.com
Port	Represents the port number dedicated to the PostgreSQL server. The database server is started using the virtual host name and port number provided. **Note:** For EnterpriseDB PostgreSQL plus Advanced Server, the default value of this attribute is 5444. Modify the value of this attribute as required. Default Value: 5432 Example: 5432

Table 3-2 lists the optional attributes for the PostgreSQL agent.

Table 3-2 Optional attributes

Attribute	Description
EnvFile	Full path of the file name to source to set the environment prior to executing the PostgreSQL commands. Symantec recommends storing the file on shared disk. The following shell environments are supported: ksh, sh, and csh. This attribute is required only if second-level monitoring is enabled. Default Value: "" Example: /postgres/data/pg.env
MonitorProgram	Absolute path name of an external, user-supplied monitor executable. For information about setting this attribute: See "Executing a customized monitoring program" on page 35. Default Value: "" Example 1.: ServerRoot/bin/myMonitor.pl Example 2.: ServerRoot/bin/myMonitor.sh arg1 arg2

Table 3-2 Optional attributes *(continued)*

Attribute	Description
SecondLevelMonitor	Used to enable second-level monitoring and specify how often it is run. Second-level monitoring is a deeper, more thorough state check of the configured PostgreSQL instance. The numeric value specifies how often the second-level monitoring routines are run. For example, if the MonitorInterval is set to 60 seconds, and the SecondLevelMonitor is set to 100, then the second level check would only get performed every 100 minutes. To provide maximum flexibility, the value set is not checked for an upper limit. You can set the second level check to occur once a month, if that is desired. Care should be taken when setting this attribute to large numbers. See "Setting up detail monitoring for the VCS agent for PostgreSQL" on page 36. **Note:** The SecondLevelMonitor attribute is applicable to VCS versions earlier than VCS 5.1 SP1 with PostgreSQL agent versions earlier than 5.1.1.0. From VCS version 5.1 SP1 with PostgreSQL agent version 5.1.1.0 onwards, the SecondLevelMonitor attribute is deprecated. Instead, a resource type level attribute LevelTwoMonitorFreq should be used to specify the frequency of in-depth monitoring. Default Value: 0 Example: 1

Table 3-2 Optional attributes *(continued)*

Attribute	Description
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 the PostgreSQL server in detail. For example, the value 5 indicates that the agent will monitor the PostgreSQL server in detail after every five online monitor intervals. **Note:** This attribute is applicable to VCS version 5.1 SP1 or later with PostgreSQL agent version 5.1.1.0 or later. If the VCS version is earlier than VCS 5.1 SP1 and the PostgreSQL agent version is earlier than 5.1.1.0, the SecondLevelMonitor attribute should be used. If you upgraded the VCS version to VCS 5.1 SP1 or later and the PostgreSQL agent version to 5.1.1.0 (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
StartOpts	The startup options for the `pg_ctl` command. Example: -l logfile
StopOpts	Shutdown options for the PostgreSQL database server. You can use this attribute to specify a shutdown mode, such as `-m fast`, which does not wait for clients to disconnect. If this attribute is not specified, the agent stops the database server with the default `-m smart` shutdown mode. For information about shutdown options, see `postgres --help`. These options are used in the `pg_ctl` command that is used while stopping the "postgres" process. Default Value: "" Example: -m fast

Table 3-2 Optional attributes *(continued)*

Attribute	Description
DBUser	A valid database user name that is used to run queries on the database during detail monitoring. This user must have privileges to run queries on or to update the table that is created for detail monitoring. Default Value: "" Example: postgres
DBName	A valid database name in which the table is created for detail monitoring. Default Value: "" Example: postgres
Table	A valid database table in the $DBUser schema on which the query is executed during detail monitoring. The table should contain a single field TSTAMP with datatype DATE. Default Value: "" Example: vcsslm

Note: For information about the additional attributes of the agent for PostgreSQL when IMF is enabled: See "Attributes that enable IMF" on page 41.

Executing a customized monitoring program

You can configure the monitor function to execute MonitorProgram. MonitorProgram is a custom monitor utility to perform a user-defined PostgreSQL server state check.

The utility is executed in the context of the UNIX user that is defined in the PostgreSQLUser attribute.

The monitor operation executes MonitorProgram if:

- The MonitorProgram attribute value is set to a valid executable utility.
- The first-level process check indicates that the PostgreSQL server instance is online.
- The SecondLevelMonitor attribute is set to 1 and the second-level check returns the server state as ONLINE.
  Or

- The SecondLevelMonitor attribute is set to greater than 1, but the second-level check is deferred for this monitoring cycle.

The monitor operation interprets the program exit code as follows:

110 or 0	PostgreSQL server is online
100 or 1	PostgreSQL server is offline
Any other value	PostgreSQL server state is unknown

Setting up detail monitoring for the VCS agent for PostgreSQL

The Symantec Cluster Server High Availability agent for PostgreSQL provides the following two levels of application monitoring:

- Primary (basic monitoring)
  In the basic monitoring mode, the agent monitors the PostgreSQL processes to verify that they are continuously active.
- Secondary (detail monitoring)
  In the detail monitoring mode, the agent executes the psql SELECT statement to monitor the health of the database.
  You can use the agent's detail monitoring capability to monitor the status of a database and listener and to increase the confidence in their availability.

Note: Disable detail monitoring before undertaking any database maintenance that involves disabling database access to external users.

Detail monitoring for a PostgreSQL resource verifies whether a database is ready for transactions by performing a SELECT transaction against a table within the database. This SELECT statement fetches the time-stamp from the table created for detail monitoring.

Ensure the following before you set up and enable detail monitoring:

- the agent is running satisfactorily at the basic level of monitoring.
- you have created a test table (with a timestamp) in the PostgreSQL database.

The example to set up detail monitoring shows how to create and test a table for use by detail monitoring, and how to enable detail monitoring.

To set up detail monitoring for PostgreSQL

1. Make the VCS configuration writable:

   ```
   haconf -makerw
   ```

2. Freeze the service group to avoid automated actions by VCS caused by an incomplete reconfiguration:

   ```
   hagrp -freeze service_group
   ```

3. Log on as a PostgreSQL user.

   ```
   su - <Owner>
   ```

4. Start the psql utility, and as the database administrator, run the following command to set up a database table:

   ```
   $ psql -h <HostName> -p <Port> -U <Admin User> -d <DBName>
   ```

   Enter the password when prompted.

5. As the database administrator, issue the following statements at the `psql` prompt to create the test table:

   ```
   CREATE USER <USER> WITH PASSWORD '<PASSWORD>';
   CREATE table <TABLE>(tstamp timestamp);
   GRANT SELECT,UPDATE ON <TABLE> TO <USER>;
   INSERT INTO <TABLE> VALUES (CURRENT_TIMESTAMP);
   \q
   ```

6. If the pg_hba.conf file is configured for password-based authentication methods, such as md5 or password, create a .pgpass file in the format:

   hostname:port:database:username:password.

   Here, use the username and password you created while creating the test table in step 5.

7. Make sure that the file permissions for the .pgpass file are 0600 or less.

8. Place the .pgpass file in the shared file system or on the local file systems. The .pgpass file must be in the same location on all nodes in the cluster.

9. Create a .env file and enter the following text in it:

   ```
   export PGPASSFILE=<PATH of .pgpass File>
   ```

   Place the .env file in the shared disk and provide the path of this .env file in the EnvFile agent attribute.

10 To test the database table for use, run the following command:

```
$ psql -A -t -h <HostName> -p <Port> -U <User> -d <DBName> -c
"select tstamp from <Table>;"
```

11 Enable detail monitoring for the PostgreSQL resource using the following VCS commands:

```
hares -modify PostgreSQLResource DBUser User

hares -modify PostgreSQLResource DBName DBName

hares -modify PostgreSQLResource Table Table

hares -modify PostgreSQLResource SecondLevelMonitor 1

haconf -dump -makero

hagrp -unfreeze service_group
```

You can also use Cluster Manager (Java Console) or Veritas Operations Manager to set these attributes.

Chapter 4

Enabling the agent for PostgreSQL to support 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 PostgreSQL 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 PostgreSQL agent for IMF, provided the following software versions are installed:

- Symantec Cluster Server (VCS) 5.1 SP1 or later

- Symantec High Availability agent for PostgreSQL version 5.1.1.0 or later

See the *Symantec Cluster Server Administrator's Guide* for more information about IMF notification module functions and 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.

Agent functions for the IMF functionality

imf_init

This function initializes the PostgreSQL agent to interface with the AMF kernel driver, which is the IMF notification module for the agent for PostgreSQL. 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

IMF

This resource type-level attribute determines whether the PostgreSQL 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 PostgreSQL supports intelligent resource monitoring for online resources only. Hence, Mode should be set to either 0 or 2.

Type and dimension: integer-association

Default values: 0 for VCS 5.1 SP1, 3 for VCS 6.0 and later.

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 to support IMF

Before you enable the PostgreSQL agent to support 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 *Symantec Cluster Server Administrator's Guide*. For details about the commands you can use to configure AMF, use the `amfconfig -h` command.

Enabling the agent to support IMF

In order to enable the PostgreSQL agent to support 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 PostgreSQL agent 5.1.1.0 installed, ensure that you run the following commands to create appropriate symbolic links:

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

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

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

If VCS is in a running state

To enable the PostgreSQL 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 PostgreSQL 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  PostgreSQL IMF -integer -assoc Mode 0 \
   MonitorFreq 1 RegisterRetryLimit 3

   # haattr -add -static PostgreSQL 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 PostgreSQL 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 PostgreSQL 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 PostgreSQL IMFRegList BaseDir DataDir
PostgreSQLUser
```

6 Save the VCS configuration.

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

7 If the PostgreSQL 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 PostgreSQL -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 PostgreSQL -sys <system>.
```

If VCS is not in a running state

To change the PostgreSQL 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[] = { BaseDir, DataDir, PostgreSQLUser }
   ```

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 PostgreSQL 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
   ```

Sample IMF configurations

An example of a type definition file for a PostgreSQL agent that is IMF-enabled is as follows. In this example, the IMF-related attributes are set to the following values:

```
AgentFile /opt/VRTSvcs/bin/Script51Agent
IMF{} { Mode=2, MonitorFreq=5, RegisterRetryLimit=3 }
IMFRegList[] { BaseDir DataDir PostgreSQLUser }
LevelTwoMonitorFreq 25

type PostgreSQL (
        static str AgentDirectory = "/opt/VRTSagents/ha/bin/PostgreSQL"
        static str AgentFile = "/opt/VRTSvcs/bin/Script50Agent"
        static str ArgList[] = { ResLogLevel, State, IState,
        PostgreSQLUser, BaseDir, DataDir, EnvFile, HostName, Port,
        StartOpts, DBUser, DBName, Table, SecondLevelMonitor,
        MonitorProgram }
        static boolean AEPTimeout = 1
        str ResLogLevel = INFO
        str PostgreSQLUser = postgres
        str HostName
        str EnvFile
        int Port = 5432
        str BaseDir
        str DataDir
        str StartOpts
        str DBUser
        str DBName
        str Table
        int SecondLevelMonitor
        str MonitorProgram
        )
```

A sample resource configuration from the /etc/VRTSvcs/conf/config/main.cf file is as follows:

```
PostgreSQL pg-sg (
        ResLogLevel = TRACE
        HostName = localhost
        EnvFile = "/server/pg.env"
        BaseDir = "/usr/local/pgsql/bin"
        DataDir = "/server"
        StartOpts = "-l /server/logfile"
```

```
DBUser = vcs_user
DBName = vcs_slm
Table = vcs_slm
)
```

Chapter 5

Configuring the service groups for PostgreSQL using the CLI

This chapter includes the following topics:

About configuring service groups for PostgreSQL

Configuring the PostgreSQL service group involves creating the PostgreSQL service group, its resources, and defining attribute values for the configured resources. You must have administrator privileges to create and configure a service group.

You can configure the service groups using one of the following:

- The Cluster Manager (Java console)
- Veritas Operations Manager
- The command-line

Before configuring the service groups for PostgreSQL

Before you configure the PostgreSQL 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 *Symantec Cluster Server Installation Guide* for more information.
- Verify that the Symantec High Availability agent for PostgreSQL is installed on all nodes in the cluster.
  See "Installing the agent in a VCS environment" on page 20.

PostgreSQL entities in a clustered environment

A service group is a logical setup containing all resources that can support a PostgreSQL instance in a clustered environment.

The required resources are as follows.

Disk group	Contains a volume and a file system, which is a mount resource containing the PostgreSQL installation files. Use the DiskGroup resource type to create this resource. Create the disk group from the shared disk so that you can import the group into any system in the cluster.
Mount	Mounts, monitors, and unmounts the file system that is dedicated to the PostgreSQL installation files. Use the Mount resource type to create this resource.
Network interface	Monitors the network interface card through which the PostgreSQL instance communicates with other services. Use the NIC resource type to create this resource.
Virtual IP	Configures the virtual IP address dedicated to the PostgreSQL instance. The external services, programs, and clients use this address to communicate with this instance. Use the IP resource type to create this resource.
PostgreSQL server	Starts, stops, and monitors the PostgreSQL server instance. Use the PostgreSQL server resource type to create this resource.

Virtualizing PostgreSQL

To ensure that your PostgreSQL machine can function properly on any node of the cluster, you need to virtualize all the parameters that could be dependent on a particular node.

Review the following basic notes for virtualization:

Host names	When installing and configuring the PostgreSQL machine, ensure that you enter the virtual host name associated with the IP address used to configure the IP resource. This ensures that if the application needs to be migrated, you are not tied down by the physical IP address given to the PostgreSQL machine.
Path names	Ensure that your application gets installed on a shared disk so that it is not constrained by anything that is local to the node. If this is not possible every time, make sure that the local data is available on each configured node.

Creating service groups for PostgreSQL under Solaris non-global zones

To configure zones on each cluster node:

1. Set up the non-global zone configuration.

   ```
   hazonesetup servicegroup_name zoneres_name zone_name password
   systems
   ```

   For example:

   ```
   hazonesetup -g servicegroup_name -r zoneres_name -z zone_name
    -p password -s systems
   ```

2. Verify the non-global zone configuration.

   ```
   hazoneverify servicegroup_name
   ```

3. Whenever you make a change that affects the zone configuration, run the `hazonesetup` command to reconfigure the zones in VCS.
4. Make sure that the zone configuration files are consistent on all nodes at all times. The file is located at /etc/zones/zone_name.xml.
5. Make sure that the application is identical on all nodes. If you update the application configuration on one node, apply the same updates to all nodes.
6. Configure the service groups for PostgreSQL.

Chapter 6

Troubleshooting the agent for PostgreSQL

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 PostgreSQL supports, see the Symantec Operations Readiness Tools (SORT) site: https://sort.symantec.com/agents.

Meeting prerequisites

Before installing the agent for PostgreSQL, double check that you meet the prerequisites.

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

See “Before you install the Symantec High Availability agent for PostgreSQL” on page 16.

Verifying virtualization

Verify that your application does not use anything that ties it down to a particular node of the cluster.

Starting the PostgreSQL server 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 PostgreSQL database server independent of the cluster framework. Refer to the cluster documentation for information about disabling a resource.

Note: Use the same parameters that the resource attributes defined within the cluster framework while restarting the resource outside the framework, like the owner of the application, the environment file etc.

- Starting the PostgreSQL server
  To start the PostgreSQL server outside cluster, execute:

  ```
  $ BaseDir/pg_ctl start -w -D DataDir -o "-p Port -h
  HostName" StartOpts
  ```

- Stopping the PostgreSQL server
  To stop the PostgreSQL server outside cluster, execute:

  ```
  $ BaseDir/pg_ctl stop -w -D DataDir
  ```

- Monitoring the PostgreSQL server
  First verify that the PostgreSQL processes are running as PostgreSQLUser. The default value is postgres.
- The agent then uses the following monitor command to verify that the PostgreSQL server is up.

  ```
  $ BaseDir/pg_ctl status -D DataDir
  ```

  Try executing this command manually to verify if the PostgreSQL server is up.

Reviewing error log files

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

The common reasons for issues are as follows:

Incorrect port, environment or parameter settings	Verify that ports have been properly configured and declared. Typically, ports from 1 through 1024 are reserved for the superuser. Ensure that parameters to the agent are correctly defined.
Expired licenses	Check the application log files for any error messages related to expired licenses. Ensure that the license keys/files have been placed at the appropriate location, as needed by the application.
Broken symlinks, missing files, and libraries	Verify your installation. Make sure that nothing is broken, and all dependencies for the executables are met.
Insufficient disk space or system parameters	Ensure that the file-system has sufficient space for creation of temporary files that the application might need. Verify that the kernel has been tuned for sufficient IPC resources, file descriptors and meets the hardware requirement. Consult your product documentation for these details.

Consult your application expert if needed.

Using trace level logging

The ResLogLevel attribute controls the level of logging that is written in a cluster log file for each PostgreSQL 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.

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 PostgreSQL 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 PostgreSQL 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 PostgreSQL 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` command to see a list of registered PIDs for a PostgreSQL resource.
  The output of the `ps -ef` command for the PostgreSQL process.

```
$ ps -ef | grep postgres postgres  4883     1  0 Aug16 ?
00:00:00 /usr/local/pgsql/bin/postgres -D /d01/pgsql/data -p 5432 -h
pg-server postgres  4893  4883  0 Aug16 ?        00:00:00 postgres:
logger process postgres  4897  4883  0 Aug16 ?   00:00:01 postgres:
writer process postgres  4898  4883  0 Aug16 ?   00:00:01 postgres:
wal writer process postgres  4899  4883  0 Aug16 ? 00:00:01 postgres:
autovacuum launcher process postgres  4900  4883  0 Aug16 ? 00:00:05
postgres:
stats collector process root     20890  3877  0 11:44 pts/1 00:00:00
grep postgres
```

  The `amfstat` command shows the PIDs monitored by the PostgreSQL Server agent.

```
# amfstat

AMF Status Report

Registered Reapers (1):
=======================
RID    PID      MONITOR    TRIGG     REAPER
0      19219       1         0       PostgreSQL

Process ONLINE Monitors (1):
=======================
RID    R_RID   PID     GROUP
1      0       4883     pg-server
```

- Run the following command to set the ResLogLevel attribute to TRACE. When you set ResLogLevel to TRACE, the agent logs messages in the PostgreSQL_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
```

Known issues

This release of the agent for PostgreSQL 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.

This results in an error message from the engine log.

Workaround

It is safe to ignore this error message.

Appendix A

Sample Configurations

This appendix includes the following topics:

About sample configurations for the agent for PostgreSQL

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 PostgreSQL. For more information about these resource types, see the *Symantec Cluster Server Bundled Agents Reference Guide*.

Sample agent type definition for PostgreSQL

VCS 5.0

```
type PostgreSQL (
    static boolean AEPTimeout = 1
    static str AgentFile = "/opt/VRTSvcs/bin/Script50Agent"
    static str AgentDirectory = "/opt/VRTSagents/ha/bin/PostgreSQL"
    static str ArgList[] = { ResLogLevel, State, IState,
    PostgreSQLUser, HostName, Port, BaseDir, DataDir, StartOpts,
    StopOpts, DBUser, DBName, Table, SecondLevelMonitor, MonitorProgram }
    str ResLogLevel = INFO
```

```
    str PostgreSQLUser = postgres
    str HostName
    str EnvFile
    int Port = 5432
    str BaseDir
    str DataDir
    str StartOpts
    str StopOpts
    str DBUser
    str DBName
    str Table
    int SecondLevelMonitor = 0
    str MonitorProgram

)
```

VCS 5.1

```
type PostgreSQL (
    static boolean AEPTimeout = 1
    static int ContainerOpts{} = { RunInContainer=1, PassCInfo=0 }
    static str AgentFile = "/opt/VRTSvcs/bin/Script50Agent"
    static str AgentDirectory = "/opt/VRTSagents/ha/bin/PostgreSQL"
    static str ArgList[] = { ResLogLevel, State, IState,
    PostgreSQLUser, BaseDir, DataDir, EnvFile, HostName, Port, StartOpts,
    StopOpts, DBUser, DBName, Table, SecondLevelMonitor, MonitorProgram }
    str ResLogLevel = INFO
    str PostgreSQLUser = postgres
    str HostName
    str EnvFile
    int Port = 5432
    str BaseDir
    str DataDir
    str StartOpts
    str StopOpts
    str DBUser
    str DBName
    str Table
    int SecondLevelMonitor = 0
    str MonitorProgram
)
```

Sample configuration files

A sample main.cf file is as follows:

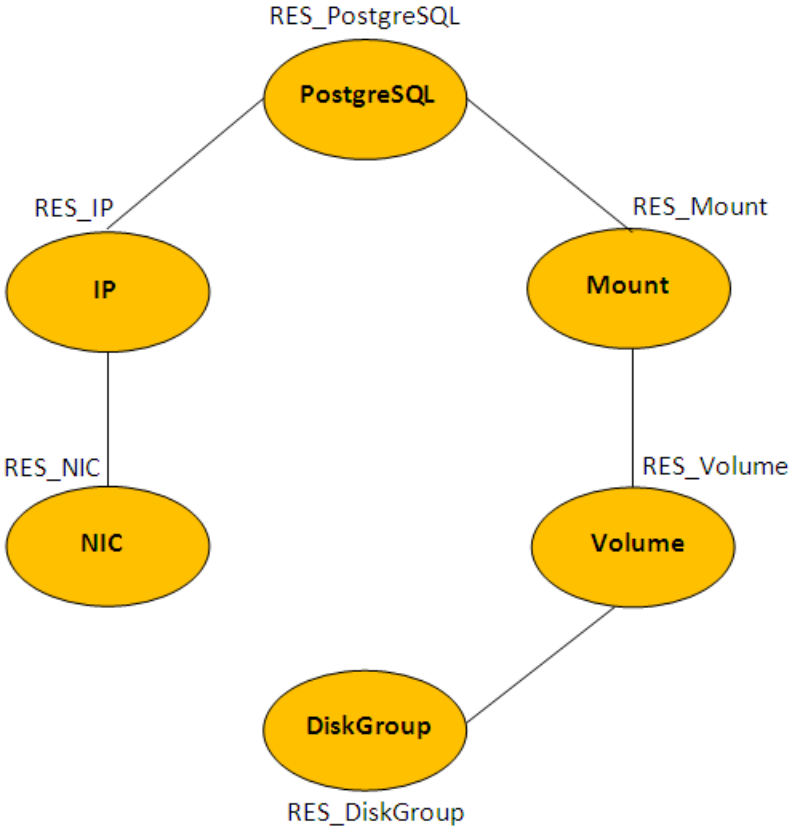
```
PostgreSQL pg_server_1 (
        Critical = 1
        ResLogLevel = TRACE
        BaseDir = "/usr/bin"
        DataDir = "/opt/postgres/data"
        HostName = pgserver
        Port = 24321
        StartOpts = "-l /tmp/pglog"
        DBUser=dbuser
        DBName=dbname
        Table=dbtable
        SecondLevelMonitor=1
        EnvFile=/var/lib/pgsql/pg.env
        )

PostgreSQL edb_pg_server_1 (
        Critical = 0
        ResLogLevel = TRACE
        PostgreSQLUser = enterprisedb
        HostName = localhost
        EnvFile = "/PostgresPlus/9.1AS/pgplus_env.sh"
        Port = 5444
        BaseDir = "/PostgresPlus/9.1AS/bin"
        DataDir = "/PostgresPlus/9.1AS/data"
        StartOpts = "-l /tmp/pglog"
        StopOpts = "-m fast"
        DBUser=dbuser
        DBName=dbname
        Table=dbtable
        SecondLevelMonitor=1
        EnvFile=/var/lib/pgsql/pg.env
        )
```

Sample service group configurations for PostgreSQL

Figure A-1 shows a service group with a PostgreSQL instance running in a VCS environment.

Figure A-1 Sample service group for a PostgreSQL instance



Index