Sun Cluster Data Services Reference Manual for Solaris OS

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

Part No: 820–2758–10
December 2007, Revision A

070925@18741

Contents

Preface

The *Sun Cluster Data Services Reference Manual* provides reference information about resources types for data services that are supplied with Sun™ Cluster software. This book is intended for experienced system administrators with extensive knowledge of Sun software and hardware. This book is not to be used as a planning or presales guide. The information in this book assumes knowledge of the Solaris™ Operating System and expertise with the volume manager software that is used with Sun Cluster software.

Both novice users and those familiar with the Solaris Operating System can use online man pages to obtain information about their SPARC™ based system or x86 based system and its features.

A man page is intended to answer concisely the question "What does this command do?" The man pages in general comprise a reference manual. They are not intended to be a tutorial.

Note – Sun Cluster software runs on two platforms, SPARC and x86. The information in this book pertains to both platforms unless otherwise specified in a special chapter, section, note, bulleted item, figure, table, or example.

Overview

The following contains a brief description of each man page section and the information it references:

- Section 4 outlines the formats of various files. The C structure declarations for the file formats are given where applicable.
- Section 5 contains miscellaneous Sun Cluster documentation such as descriptions of resource types.

The following is a generic format for man pages. The man pages of each manual section generally follow this order, but include only needed headings. For example, if no bugs can be reported, no BUGS section is included. See the `intro` pages for more information and detail about each section, and `man`(1) for general information about man pages.

NAME — This section gives the names of the commands or functions that are documented, followed by a brief description of what they do.

SYNOPSIS — This section shows the syntax of commands or functions. If a command or file does not exist in the standard path, its full path name is shown. Options and arguments are alphabetized, with single-letter arguments first, and options with arguments next, unless a different argument order is required.

The following special characters are used in this section:

- [ ] — Brackets. The option or argument that is enclosed in these brackets is optional. If the brackets are omitted, the argument must be specified.
- . . . — Ellipses. Several values can be provided for the previous argument, or the previous argument can be specified multiple times, for example, "`filename...`".
- | — Separator. Only one of the arguments separated by this character can be specified at a time.
- { } — Braces. The options and/or arguments enclosed within braces are interdependent. All characters within braces must be treated as a unit.

PROTOCOL — This section occurs only in subsection 3R and indicates the protocol description file.

DESCRIPTION — This section defines the functionality and behavior of the service. Thus it describes concisely what the command does. DESCRIPTION does not discuss OPTIONS or cite EXAMPLES. Interactive commands, subcommands, requests, macros, and functions are described under USAGE.

IOCTL — This section appears on pages in Section 7 only. Only the device class that supplies appropriate parameters to the `ioctl`(2) system call is called `ioctl` and generates its own heading. `ioctl` calls for a specific device are listed alphabetically (on the man page for that specific device).

	`ioctl` calls are used for a particular class of devices. All these calls have an `io` ending, such as `mtio`(7I).
OPTIONS	This section lists the command options with a concise summary of what each option does. The options are listed literally and in the order they appear in the SYNOPSIS section. Possible arguments to options are discussed under the option, and where appropriate, default values are supplied.
OPERANDS	This section lists the command operands and describes how they affect the actions of the command.
OUTPUT	This section describes the output – standard output, standard error, or output files – generated by the command.
RETURN VALUES	If the man page documents functions that return values, this section lists these values and describes the conditions under which they are returned. If a function can return only constant values, such as 0 or –1, these values are listed in tagged paragraphs. Otherwise, a single paragraph describes the return values of each function. Functions that are declared void do not return values, so they are not discussed in RETURN VALUES.
ERRORS	On failure, most functions place an error code in the global variable `errno` that indicates why they failed. This section lists alphabetically all error codes a function can generate and describes the conditions that cause each error. When more than one condition can cause the same error, each condition is described in a separate paragraph under the error code.
USAGE	This section lists special rules, features, and commands that require in-depth explanations. The subsections that are listed here are used to explain built-in functionality: Commands Modifiers Variables Expressions Input Grammar
EXAMPLES	This section provides examples of usage or of how to use a command or function. Wherever possible, a complete

	example, which includes command-line entry and machine response, is shown. Whenever an example is given, the prompt is shown as `example%`, or if the user must be superuser, `example#`. Examples are followed by explanations, variable substitution rules, or returned values. Most examples illustrate concepts from the SYNOPSIS, DESCRIPTION, OPTIONS, and USAGE sections.
ENVIRONMENT VARIABLES	This section lists any environment variables that the command or function affects, followed by a brief description of the effect.
EXIT STATUS	This section lists the values the command returns to the calling program or shell and the conditions that cause these values to be returned. Usually, zero is returned for successful completion, and values other than zero are returned for various error conditions.
FILES	This section lists all file names that are referred to by the man page, files of interest, and files created or required by commands. Each file name is followed by a descriptive summary or explanation.
ATTRIBUTES	This section lists characteristics of commands, utilities, and device drivers by defining the attribute type and its corresponding value. See `attributes`(5) for more information.
SEE ALSO	This section lists references to other man pages, in-house documentation, and outside publications.
DIAGNOSTICS	This section lists diagnostic messages with a brief explanation of the condition that caused the error.
WARNINGS	This section lists warnings about special conditions that could seriously affect your working conditions. WARNINGS is not a list of diagnostics.
NOTES	This section lists additional information that does not belong anywhere else on the page. NOTES covers points of special interest to the user. Critical information is never covered here.
BUGS	This section describes known bugs and, wherever possible, suggests workarounds.

REFERENCE

SC32DS 4

Name custom_action_file – file that defines custom behavior of fault monitors for HA Oracle server resources and Oracle 9i RAC server resources

Description A custom action file is a plain text file. The file contains one or more entries that define the custom behavior of fault monitors for the following resources:

- **HA Oracle server resources.** These resources are instances of the SUNW.oracle_server(5) resource type.
- **Oracle 9i Real Application Clusters (RAC) server resources.** These resources are instances of the SUNW.scalable_rac_server(5) resource type.

Each entry defines the custom behavior for a single database management system (DBMS) error, a single timeout error, or several logged alerts. A maximum of 1024 entries is allowed in a custom action file.

Note – Each entry in a custom action file overrides the preset action for an error, or specifies an action for an error for which no action is preset. Create entries in a custom action file *only* for the preset actions that you are overriding or for errors for which no action is preset. Do *not* create entries for actions that you are not changing.

An entry in a custom action file consists of a sequence of keyword-value pairs that are separated by semicolons. Each entry is enclosed in braces.

The format of an entry in a custom action file is as follows:

```
{
[ERROR_TYPE=DBMS_ERROR|SCAN_LOG|TIMEOUT_ERROR;]
ERROR=error-spec;
[ACTION=SWITCH|RESTART|STOP|NONE;]
[CONNECTION_STATE=co|di|on|*;]
[NEW_STATE=co|di|on|*;]
[MESSAGE="message-string"]
}
```

White space may be used between separated keyword-value pairs and between entries to format the file.

The meaning and permitted values of the keywords in a custom action file are as follows:

`ERROR_TYPE`
Indicates the type of the error that the server fault monitor has detected. The following values are permitted for this keyword:

`DBMS_ERROR`	Specifies that the error is a DBMS error.
`SCAN_LOG`	Specifies that the error is an alert that is logged in the alert log file.
`TIMEOUT_ERROR`	Specifies that the error is a timeout.

The ERROR_TYPE keyword is optional. If you omit this keyword, the error is assumed to be a DBMS error.

ERROR

Identifies the error. The data type and the meaning of *error-spec* are determined by the value of the ERROR_TYPE keyword as shown in the following table.

ERROR_TYPE	Data Type	Meaning
DBMS_ERROR	Integer	The error number of a DBMS error that is generated by Oracle
SCAN_LOG	Quoted regular expression	A string in an error message that Oracle has logged to the Oracle alert log file
TIMEOUT_ERROR	Integer	The number of consecutive timed-out probes since the server fault monitor was last started or restarted

You must specify the ERROR keyword. If you omit this keyword, the entry in the custom action file is ignored.

ACTION

Specifies the action that the server fault monitor is to perform in response to the error. The following values are permitted for this keyword:

NONE — Specifies that the server fault monitor ignores the error.

STOP — Specifies that the server fault monitor is stopped.

RESTART — Specifies an action that depends on the type of resource for which the fault monitor that is being customized:

- **HA Oracle server resource.** Specifies that the server fault monitor stops and restarts the entity that is specified by the value of the Restart_type extension property of the SUNW.oracle_server resource.
- **Oracle 9i RAC server resource.** Specifies that the server fault monitor stops and restarts the Oracle 9i RAC server resource.

SWITCH — Specifies that the server fault monitor switches over the database server resource group to another node.

Note – Do *not* specify the SWITCH keyword in the custom action file for an Oracle 9i RAC server fault monitor. For the Oracle 9i RAC server fault monitor, the SWITCH keyword performs no action.

The `ACTION` keyword is optional. If you omit this keyword, the server fault monitor ignores the error.

`CONNECTION_STATE`
Specifies the required state of the connection between the database and the server fault monitor when the error is detected. The entry applies only if the connection is in the required state when the error is detected. The following values are permitted for this keyword:

`*` Specifies that the entry always applies, regardless of the state of the connection.

`co` Specifies that the entry applies only if the server fault monitor is attempting to connect to the database.

`on` Specifies that the entry applies only if the server fault monitor is online. The server fault monitor is online if it is connected to the database.

`di` Specifies that the entry applies only if the server fault monitor is disconnecting from the database.

The `CONNECTION_STATE` keyword is optional. If you omit this keyword, the entry always applies, regardless of the state of the connection.

`NEW_STATE`
Specifies the state of the connection between the database and the server fault monitor that the server fault monitor must attain after the error is detected. The following values are permitted for this keyword:

`*` Specifies that the state of the connection must remain unchanged.

`co` Specifies that the server fault monitor must disconnect from the database and reconnect immediately to the database.

`di` Specifies that the server fault monitor must disconnect from the database. The server fault monitor reconnects when it next probes the database.

The `NEW_STATE` keyword is optional. If you omit this keyword, the state of the database connection remains unchanged after the error is detected.

`MESSAGE`
Specifies an additional message that is printed to the resource's log file when this error is detected. The message must be enclosed in double quotes. This message is additional to the standard message that is defined for the error.

The `MESSAGE` keyword is optional. If you omit this keyword, no additional message is printed to the resource's log file when this error is detected.

Examples

EXAMPLE 1 Changing the Response to a DBMS Error to Restart

```
{
ERROR_TYPE=DBMS_ERROR;
```

EXAMPLE 1 Changing the Response to a DBMS Error to Restart *(Continued)*

```
ERROR=4031;
ACTION=restart;
CONNECTION_STATE=*;
NEW_STATE=*;
MESSAGE="Insufficient memory in shared pool.";
}
```

This example shows an entry in a custom action file that overrides the preset action for DBMS error 4031. This entry specifies the following behavior:

- In response to DBMS error 4031, the action that the server fault monitor performs is restart.
- This entry applies regardless of the state of the connection between the database and the server fault monitor when the error is detected.
- The state of the connection between the database and the server fault monitor must remain unchanged after the error is detected.
- The following message is printed to the resource's log file when this error is detected:

  ```
  Insufficient memory in shared pool.
  ```

EXAMPLE 2 Ignoring a DBMS Error

```
{
ERROR_TYPE=DBMS_ERROR;
ERROR=4030;
ACTION=none;
CONNECTION_STATE=*;
NEW_STATE=*;
MESSAGE="";
}
```

This example shows an entry in a custom action file that overrides the preset action for DBMS error 4030. This entry specifies the following behavior:

- The server fault monitor ignores DBMS error 4030.
- This entry applies regardless of the state of the connection between the database and the server fault monitor when the error is detected.
- The state of the connection between the database and the server fault monitor must remain unchanged after the error is detected.
- No additional message is printed to the resource's log file when this error is detected.

EXAMPLE 3 Changing the Response to a Logged Alert

```
{
ERROR_TYPE=SCAN_LOG;
ERROR="ORA-00600: internal error";
ACTION=RESTART;
}
```

This example shows an entry in a custom action file that overrides the preset action for logged alerts about internal errors. This entry specifies the following behavior:

- In response to logged alerts that contain the text `ORA-00600: internal error`, the action that the server fault monitor performs is restart.
- This entry applies regardless of the state of the connection between the database and the server fault monitor when the error is detected.
- The state of the connection between the database and the server fault monitor must remain unchanged after the error is detected.
- No additional message is printed to the resource's log file when this error is detected.

EXAMPLE 4 Changing the Maximum Number of Consecutive Timed-Out Probes

```
{
ERROR_TYPE=TIMEOUT;
ERROR=2;
ACTION=NONE;
CONNECTION_STATE=*;
NEW_STATE=*;
MESSAGE="Timeout #2 has occurred.";
}

{
ERROR_TYPE=TIMEOUT;
ERROR=3;
ACTION=NONE;
CONNECTION_STATE=*;
NEW_STATE=*;
MESSAGE="Timeout #3 has occurred.";
}

{
ERROR_TYPE=TIMEOUT;
ERROR=4;
ACTION=NONE;
CONNECTION_STATE=*;
NEW_STATE=*;
MESSAGE="Timeout #4 has occurred.";
}
```

EXAMPLE 4 Changing the Maximum Number of Consecutive Timed-Out Probes *(Continued)*

```
{
ERROR_TYPE=TIMEOUT;
ERROR=5;
ACTION=RESTART;
CONNECTION_STATE=*;
NEW_STATE=*;
MESSAGE="Timeout #5 has occurred. Restarting.";
}
```

This example shows the entries in a custom action file for increasing the maximum number of consecutive timed-out probes to five. These entries specify the following behavior:

- The server fault monitor ignores the second consecutive timed-out probe through the fourth consecutive timed-out probe.
- In response to the fifth consecutive timed-out probe, the action that the server fault monitor performs is restart.
- The entries apply regardless of the state of the connection between the database and the server fault monitor when the timeout occurs.
- The state of the connection between the database and the server fault monitor must remain unchanged after the timeout occurs.
- When the second consecutive timed-out probe through the fourth consecutive timed-out probe occurs, a message of the following form is printed to the resource's log file:

  `Timeout #`*`number`* `has occurred.`
- When the fifth consecutive timed-out probe occurs, the following message is printed to the resource's log file:

  `Timeout #5 has occurred. Restarting.`

See Also SUNW.oracle_server(5), SUNW.scalable_rac_server(5)

Sun Cluster Data Service for Oracle Guide for Solaris OS, *Sun Cluster Data Service for Oracle RAC Guide for Solaris OS*

REFERENCE

SC32DS 5

Name SUNW.apache, apache – resource type implementation for failover and scalable Apache Web Server

Description The Apache Web Server data service for Sun Cluster 3.2 is configured as a resource managed by the Sun Cluster Resource Group Manager (RGM).

You must set the following properties on an Apache resource by using `clresource`(1CL).

Standard Properties The standard resource properties `Scalable`, `Network_resources_used`, `Port_list`, `Load_balancing_policy`, and `Load_balancing_weights` are common to all scalable resource types.

The `SUNW.apache` resource type supports two modes. The first mode is a scalable mode that exploits the cluster networking facility to permit the Apache resource to run on multiple nodes simultaneously. The second mode is a failover mode, in which the Apache resource runs on only one node at a time. The `Scalable` property is set at resource creation time to indicate the mode in which the service operates. The default is `FALSE` (failover mode).

See`r_properties`(5) for a complete description of the following resource properties.

`Load_balancing_policy`

Default `LB_WEIGHTED`

Tunable At creation

`Load_balancing_weights`

Default `NULL`

Tunable Any time

`Network_resources_used`

Default `No default`

Tunable At creation

`Port_list`

Default `80/tcp`

Tunable At creation

`Retry_count`

Default `2`

Tunable Any time

`Retry_interval`

Default `300`

Tunable Any time

`Thorough_probe_interval`

Default `60`

Tunable Any time

Extension Properties `Bin_dir`
Type string. Indicates the location of Apache Web server binaries. You must specify this property at resource creation time.

`Monitor_retry_count`
Type integer. Default is 4. Controls the restarts of the fault monitor. This property indicates the number of times the fault monitor is restarted by the process monitor facility and corresponds to the `-n` option passed to the`pmfadm`(1M) command. The number of restarts is counted in a specified time window (see the property `Monitor_retry_interval`). Note that this property refers to the restarts of the fault monitor itself, not the web server. The restarts of the web server are controlled by the system-defined properties `Thorough_Probe_Interval`, `Retry_Interval`, and `Retry_Count`, as specified in their descriptions. See `clresource`(1CL). You can modify the value for this property at any time.

`Monitor_retry_interval`
Type integer. Default is 2. Indicates the time in minutes, over which the failures of the fault monitor are counted, and corresponds to the `-t` option passed to the`pmfadm`(1M) command. If the number of times the fault monitor fails exceeds the value of `Monitor_retry_count`, the fault monitor is not restarted by the process monitor facility. You can modify the value for this property at any time.

`Monitor_Uri_List`
Type string array. Default is " ". Introduced in release 3.1 10/03. This property enables you to ensure that application components are responding by querying the configured URIs. The `Monitor_Uri_List` property is used for detailed fault monitoring of Sun Cluster HA for Apache Web Server. The fault monitor periodically runs the `HTTP GET` command for the URIs. The monitor takes action if the HTTP request returns with response code 500 "Internal Server Error" or if the application server does not respond. An example URI setting is `http://logical-hostname/App/tester`. If the configured URIs are implemented by using a servlet in the web server, detailed monitoring of the web server Java Virtual Machine (JVM) is possible.

`Probe_timeout`
Type integer. Defaults to 90. This property is the time-out value (in seconds) used by the fault monitor to probe an Apache instance. You can modify the value for this property at any time.

Examples

EXAMPLE 1 Creating a Failover Apache Resource

For this example to work, the data service must first be installed. This example creates a failover Apache resource named `apache-failover` in an existing resource group named `web-rg`. `web-rg` is assumed to contain a `LogicalHostname` resource, which identifies the

EXAMPLE 1 Creating a Failover Apache Resource *(Continued)*

logical hostname associated with the resource group. Another assumption is that the `Port_list` property defaults to `80/tcp`, that is, the Apache instance is listening on port 80.

```
example# clresourcetype register SUNW.apache
example# clresource create -g web-rg -t SUNW.apache \
 -p Bin_dir=/global/apache/https-web/bin apache-failover
```

In this example, the Apache resource created is named `apache-failover`, which listens on port 80, with a corresponding Apache instance in the directory `/global/apache/https-web`.

EXAMPLE 2 Creating a Scalable Apache Resource

For this example to work, the data service must first be installed. This example creates a scalable Apache resource named `apache-scalable` in a resource group named `web-rg ,` which is configured to run simultaneously on all four nodes of a four-node cluster. The `apache-scalable` resource is configured to listen on port 8080 and uses the IP addresses as configured in a `SharedAddress` resource named `www_foo_com`, which is contained in the resource group `foo_com_RG`.

```
example# clresourcegroup create -p Maximum_primaries=4 \
-p Desired_primaries=4 -p RG_dependencies=foo_com_RG web-rg
example# clresourcetype register SUNW.apache
example# clresource create -g web-rg -t SUNW.apache \
-p Bin_dir=/global/apache/https-web/bin \
-p Port_list=8080/tcp -y Scalable=TRUE \
-p Network_resources_used=www_foo_com apache-scalable
```

EXAMPLE 3 Setting `Monitor_uri_list` for Scalable Sun Cluster HA for Apache Instance

The following example shows how to set the `Monitor_uri_list` extension property when a scalable Sun Cluster HA for Apache instance is added to a configuration. The `Monitor_uri_list` extension property is not supported with a secure Sun Cluster HA for Apache instance.

```
(Add an insecure Apache instance with default load balancing.)

example# clresource create -g resource-group-1 \
-t SUNW.apache -p Bin_dir=/opt/apache/bin \
-p Monitor_Uri_list=http://schost-1:8000/servlet/monitor \
-p Network_resources_used=schost-1,... \
-p Scalable=True -p Port_list=8000/tcp apache-insecure-1
```

EXAMPLE 4 Setting `Monitor_uri_list` for Failover Sun Cluster HA for Apache Instance

The following example shows how to set the `Monitor_uri_list` extension property when a failover Sun Cluster HA for Apache instance is added to a configuration. The `Monitor_uri_list` extension property is not supported with a secure Sun Cluster HA for Apache instance.

```
(Add an insecure Apache application resource instance.)

# clresource create -g resource-group-1 \
-t SUNW.apache -p Bin_dir=/opt/apache/bin \
-p Monitor_Uri_list=http://schost-1:80/servlet/monitor \
-p Network_resources_used=schost-1 \
-p Scalable=False -p Port_list=80/tcp apache-insecure-1
```

Files

`Bin_dir/apachectl`
The `apachectl start` command is used by HA-Apache to start a non-SSL Apache web server.

The `apachectl startssl` command is used by HA-Apache to start an Apache web server that uses `mod_ssl`.

`Bin_dir/httpsdctl`
The `httpsdctl start` command is used by HA-Apache to start an Apache-ssl web server.

`Bin_dir/keypass`
This file must be created for an Apache web server that uses `mod_ssl` for handling `https` requests. Only the owner should have read, write, or execute permissions to this file. All other users must not have permissions to this file.

If the web server does not use encrypted private keys, the contents of this file are irrelevant. For a web server that uses encrypted private keys, this file is called during resource startup with `host:port` and `algorithm` as its two arguments. The web server expects the pass phrase for the key corresponding to that host and port combination to be printed to `stdout`.

For example, for a secure web server listening on ports 8080 and 8888 that use RSA encrypted private keys for both ports, the `keypass` file could look like the following:

```
#!/bin/ksh
host=‘echo $1 | cut -d: -f1‘
port=‘echo $1 | cut -d: -f2‘
algorithm=$2

if [ "$host" = "button-1.eng.sun.com" \\
    -a "$algorithm" = "RSA" ]; then
case "$port" in
    8080) echo passphrase-for-8080;;
    8888) echo passphrase-for-8888;;
```

```
esac
fi
```

Attributes See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWscapc

See Also `pmfadm`(1M) `scha_resource_get`(3HA), `clresourcetype`(1CL), `clresourcegroup`(1CL), `attributes`(5), `r_properties`(5), `scalable_service`(5)

Sun Cluster Data Service for Apache Guide for Solaris OS,*Sun Cluster Data Services Planning and Administration Guide for Solaris OS*

Name SUNW.dns, dns – resource type implementation for failover Domain Name Service (DNS)

Description The DNS data service for Sun Cluster 3.2 is configured as a resource managed by the Sun Cluster Resource Group Manager (RGM). You must set the following properties on a DNS resource.

See `r_properties`(5) for a complete description of the following resource properties.

Standard Properties

`Network_resources_used`

Default `No default`

Tunable At creation

`Port_list`

Default `53/udp`

Tunable At creation

For DNS resources, the value of `53/udp` is the only recommended value.

`Retry_count`

Default `2`

Tunable When disabled

`Retry_interval`

Default `300`

Tunable When disabled

`Thorough_probe_interval`

Default `60`

Tunable Any time

Extension Properties

`Confdir_list`

Type string array. This property is the path name to the configuration directory that contains the file `named.conf` of the DNS resource. You must specify only one value for this property at resource creation time.

`DNS_mode`

Type string array. This property is the configuration file to be used for starting DNS. The default is `conf`, which means that the DNS is started by using the `named.conf` file located in the directory pointed to by the value of the `Confdir_list` property. A value of `boot` means that DNS is started with the `named.boot` file as the configuration file. You can specify the value of this property at resource creation time only.

`Monitor_retry_count`
Type integer. Default is `4`. This property controls the restarts of the fault monitor. It indicates the number of times the fault monitor is restarted by the process monitor facility and corresponds to the `-n` option passed to the `pmfadm`(1M) command. The number of restarts is counted in a specified time window (see the property `Monitor_retry_interval`). Note that this property refers to the restarts of the fault monitor itself, not DNS. The restarts of DNS are controlled by the system-defined properties `Thorough_Probe_Interval` and `Retry_Intervalan`d`Retry_Count`, as specified in the description of those system-defined properties. See `clresource`(1CL). You can modify the value for this property any time.

`Monitor_retry_interval`
Type integer. Default is `2`. Indicates the time (in minutes) over which the failures of the fault monitor are counted and corresponds to the `-t` option passed to the `pmfadm`(1M) command. If the number of times the fault monitor fails exceeds the extension property `Monitor_retry_count`, the fault monitor is not restarted by the Process Monitor Facility. You can modify the value for this property any time.

`Probe_timeout`
Type integer. Default is `120`. Indicates the time-out value (in seconds) used by the fault monitor to probe a DNS instance. You can modify the value for this property any time.

Examples

EXAMPLE 1 Initiating a Failover DNS Resource

For this example to work, you must first install the data service. This example instantiates a failover DNS resource named `dnss` in a resource group named `dns-rg`. `dns-rg` is assumed to contain at least one `LogicalHostname` resource, which identifies the logical hostnames associated with the resource group.

```
example# clresourcetype register SUNW.dns
example# clresource create -g dns-rg –t SUNW.dns \
-p Confdir_list=/global/dns dnss
```

In this example, the DNS resource created is named `dnss`, which listens on port 53, with a corresponding configuration directory path name `/global/dns`. The configuration file that is used for starting the DNS resource is `named.conf`, located under `/global/dns`.

EXAMPLE 2 Instantiating a Failover DNS Resource

For this example to work, the data service must first be installed. This example instantiates a failover DNS resource named `dns` in a resource group named `dns-rg`, which uses the `LogicalHostname` resource `lh-specific`.

```
example# clresourcetype register SUNW.dns
example# clresource create –g dns-rg –t SUNW.dns \
-p Confdir_list=/global/dns \
-p Network_resources_used=lh-specific dns-lh
```

EXAMPLE 2 Instantiating a Failover DNS Resource *(Continued)*

In this example, the `LogicalHostname` resource `lh-specific` must be a resource in the `dns-rg` resource group.

Attributes

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWscdns

See Also

`in.named`(1M), `pmfadm`(1M), `scha_resource_get`(3HA), `clresourcetype`(1CL), `clresourcegroup`(1CL), `named.conf`(4), `attributes`(5), `r_properties`(5)

Sun Cluster Data Services for DNS Guide for Solaris OS

Name SUNW.hadb, hadb – resource type implementation for Sun Java System Application Server EE (HADB) that is mastered on multiple nodes simultaneously

Description The `SUNW.hadb` resource type represents the Sun Java System Application Server EE (HADB) component in a Sun Cluster configuration.

You must set the following properties on a `SUNW.hadb` resource by using `clresource`(1CL).

Standard Properties See `r_properties`(5) for a complete description of the following resource properties.

`Thorough_probe_interval`

Default `180`

Minimum `120`

Tunable `Any time`

`Stop_timeout`

Default `180`

Minimum `60`

Tunable `Any time`

`Start_timeout`

Default `600`

Tunable `Any time`

Extension Properties `Confdir_list`
Type string array. This property is a path name set to the Sun Java System Application Server EE (HADB) configuration directory for the HADB database. For example, `/etc/opt/SUNWhadb/dbdef/hadb`. This property is tunable when the resource is disabled.

`DB_Name`
Type string. No default. This property contains the name of the HADB database. This property is tunable when the resource is disabled.

`HADB_ROOT`
Type string. No default. This property is a path name set to the Sun Java System Application Server EE (HADB) application binary location. For example, `/opt/SUNWappserver7/SUNWhadb/4`. This property is tunable when the resource is disabled.

`Auto_recovery`
Type boolean. Default is FALSE. If the `Auto_recovery` extension property is set to `TRUE` and the agent is unable to start the database, the HADB data service attempts to recover the database by reinitializing it. The data service reinitializes the database by running the `hadbm clear -- fast` command, and then running the command specified by the `Auto_recovery_command` extension property. This property is tunable at any time.

`Auto_recovery_command`
Type string. Default is "". This property specifies the command to be run when the HADB data service recovers the database by reinitializing the database. The HADB data service runs the command after clearing the database. This property is tunable at any time.

`DB_password_file`
Type string. Default is "". This property specifies the file that contains the password to be used for the system user of the database. The `hadbm` commands that require a password will have the `DB_password_file` extension property passed as the value of the `hadbm --dbpasswordfile` command. This argument is needed if the `Auto_recovery` extension property is set to `TRUE` because the `hadbm –-clear` command requires a password. This property is tunable at any time.

Examples

The following examples show the creation of a resource group for a Sun Java System Application Server EE (HADB) resource and the creation of a resource in this resource group. The examples that show the creation of a Sun Java System Application Server EE (HADB) resource illustrate alternative configurations of the Sun Java System Application Server EE (HADB) resource.

The examples assume that the `SUNWschadb` package is already installed.

EXAMPLE 1 Creating a Resource Group for a Sun Java System Application Server EE (HADB) Resource

This example shows the creation of a resource group for a Sun Java System Application Server EE (HADB) resource on a six-node cluster. Sun Java System Application Server EE (HADB) is mastered on all nodes in the cluster. The resource group is named `hadb-rg`.

```
# clresourcegroup create \
-p maximum_primaries=6 -p desired_primaries=6 hadb-rg
```

EXAMPLE 2 Creating a Sun Java System Application Server EE (HADB) Resource Without `Auto_recovery`

This example shows the creation of a Sun Java System Application Server EE (HADB) resource without `Auto_recovery` that is named `hadb-rs`. This resource is created in an existing resource group that is named `hadb-rg`. The creation of the `hadb-rg` resource group is shown in the example for the creation of a resource group for a Sun Java System Application Server EE (HADB) resource.

```
# clresource create -g hadb-rg -t SUNW.hadb \
-p confdir_list=/etc/opt/SUNWhadb/dbdef/hadb \
-p hadb_root=/opt/SUNWappserver7/SUNWhadb/4
\-p db_name=hadb hadb-rs
```

EXAMPLE 3 Creating a Sun Java System Application Server EE (HADB) Resource With `Auto_recovery`

This example shows the creation of a Sun Java System Application Server EE (HADB) resource with `Auto_recovery` that is named `hadb-rs`. This resource is created in an existing resource group that is named `hadb-rg`. The creation of the `hadb-rg` resource group is shown in the example for the creation of a resource group for a Sun Java System Application Server EE (HADB) resource.

```
# clresource create -g hadb-rg -t SUNW.hadb \
-p confdir_list=/etc/opt/SUNWhadb/dbdef/hadb \
-p hadb_root=/opt/SUNWappserver7/SUNWhadb/4 -p db_name=hadb \
-p auto_recovery=true \
-p auto_recovery_command=/usr/local/etc/create-session-store \
-p db_password_file=/usr/local/etc/hadb-password-file hadb-rs
```

Attributes

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWschadb

See Also

`clresourcetype`(1CL), `clresourcegroup`(1CL), `scha_resource_get`(3HA), `pmfadm`(1M), `attributes`(5), `r_properties`(5), `scalable_service`(5)

Sun Cluster Data Service for Sun Java System Application Server EE (HADB) for Solaris OS, *Sun Cluster Data Services Planning and Administration Guide for Solaris OS*

Name SUNW.hadb_ma, hadb_ma – resource type implementation for Sun Java System Application Server EE (HADB) (hadb_ma)

Description The `SUNW.hadb_ma` resource type represents the Sun Java System Application Server EE (HADB version 4.4) application in a Sun Cluster configuration.

Standard properties and extension properties that are defined for the `SUNW.hadb_ma` resource type are described in the subsections that follow. To set these properties for an instance of the `SUNW.hadb_ma` resource type, use the `clresource`(1CL) command.

Standard Properties See `r_properties`(5) for a complete description of the following resource properties.

`Network_resources_used`

Default Null string

Tunable When disabled

`Retry_count`

Maximum 10

Default 2

Tunable Any time

`Retry_interval`

Maximum 3600

Default 480

Tunable Any time

`Thorough_probe_interval`

Maximum 3600

Default 120

Tunable Any time

Extension Properties The extension properties of this resource type are as follows:

`HADB_MA_CFG`
The full path to the configuration file that is used to start the HADB Management Agent Server.

Data type String

Default `/etc/opt/SUNWhadb/mgt.cfg`

Range Not applicable

Tunable When disabled

`HADB_MA_START`
The full path to the script that is used to start and stop the HADB Management Agent Server. This script must be able to start and stop the MA Server without any input from the user. Any configuration parameters must be specified in the file that is indicated by the extension property `HADB_MA_CFG`.

Data type String

Default `/etc/init.d/ma-initd`

Range Not applicable

Tunable When disabled

`HADB_MA_USER`
The user name of the user who starts the HADB Management Agent Server.

Data type String

Default `root`

Range Not applicable

Tunable When disabled

`HADB_ROOT`
The complete path to the HADB installation directory. This directory contains the directory struction `bin/`, which contains the files `ma` and `hadbm`.

Data type String

Default `/opt/SUNWhadb/4`

Range Not applicable

Tunable When disabled

`HADBM_PASSWORDFILE`
The complete path to the file that contains the HADBM administrative password. This property must be set if the HADB management domain is created with an administrative password.

Data type String

Default Null

Range Not applicable

Tunable Any time

Examples

EXAMPLE 1 Creating a Resource for `SUNW.hadb_ma`

For this example to work, the Sun Cluster HA for Sun Java System Application Server EE (HADB) data service must first be installed. This data service includes all the packages to make Sun Java System Application Server EE (HADB) highly available.

This example creates an HADB MA resource mastered on multiple nodes. The resource is created in an existing resource group, which is assumed to be mastered on multiple nodes. To create this resource the following commands are run:

```
# clresourcetype register SUNW.hadb_ma
# clresource create -g hadb-ma-rg -t SUNW.hadb_ma hadb-ma-rs
```

In this example, the resource group is named `hadb-ma-rg` and the resource is named `hadb-ma-rs`. The default values are used for the extension properties.

Attributes

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWschadb_ma

See Also

`clresourcetype`(1CL), `clresourcegroup`(1CL), `scha_resource_get`(3HA), `pmfadm`(1M), `attributes`(5), `r_properties`(5), `scalable_service`(5)

Sun Cluster Data Service for Sun Java System Application Server EE (HADB) Guide for Solaris OS and *Sun Cluster Data Services Planning and Administration Guide for Solaris OS*

Name SUNW.iws, iws – resource type implementation for failover and scalable Sun Java System Web Server

Description The `SUNW.iws` resource type represents the Sun Java System Web Server application in a Sun Cluster configuration.

You must set the following properties on an `SUNW.iws` resource by using `clresource`(1CL).

Standard Properties The standard resource properties `Scalable`, `Network_resources_used`, `Port_list`, `Load_balancing_policy`, and `Load_balancing_weights` are common to all scalable resource types.

The `SUNW.iws` resource type supports two modes. The first mode is a scalable mode that exploits the cluster networking facility to permit the `iWS` resource to run on multiple nodes simultaneously. The second mode is a failover mode, in which the `iWS` resource runs on only one node at a time. The `Scalable` property is set at resource creation time to indicate the mode in which the service operates. The default is `FALSE` (failover mode).

See `r_properties`(5) for a complete description of the following resource properties.

`Load_balancing_policy`

Default `LB_WEIGHTED`

Tunable `At creation`

`Load_balancing_weights`

Default `NULL`

Tunable Any time

`Network_resources_used`

Default `No default`

Tunable At creation

`Port_list`

Default `80/tcp`

Tunable At creation

`Retry_count`

Default `2`

Tunable Any time

`Retry_interval`

Default `300`

Tunable Any time

`Thorough_probe_interval`

Default `60`

Tunable Any time

Extension Properties `Confdir_list`

Type string array. This property is a comma-separated list of path names. Each element in the list is the path name of an `iWS` instance directory. If an `iWS` instance is in secure mode, then the directory must contain a file named `keypass`, which contains the secure key password needed to start this instance. You must specify this property at resource creation time.

`Monitor_retry_count`

Type integer. Default is `4`. Controls the restarts of the fault monitor. This property indicates the number of times the fault monitor is restarted by the process monitor facility and corresponds to the `-n` option passed to the `pmfadm`(1M) command. The number of restarts is counted in a specified time window (see the property `Monitor_retry_interval`). Note that this property refers to the restarts of the fault monitor itself, not the web server. The restarts of the web server are controlled by the system-defined properties `Thorough_Probe_Interval`, `Retry_Interval`, and `Retry_Count`, as specified in their descriptions. See `clresource`(1CL). You can modify the value for this property at any time.

`Monitor_retry_interval`

Type integer. Default is `2`. Indicates the time in minutes, over which the failures of the fault monitor are counted, and corresponds to the `-t` option passed to the `pmfadm`(1M) command. If the number of times the fault monitor fails exceeds the value of `Monitor_retry_count`, the fault monitor is not restarted by the Process Monitor Facility. You can modify the value for this property at any time.

`Monitor_Uri_List`

Type string array. Default is ""; introduced in release 3.1 10/03. This property allows you to ensure that application components are responding by querying the configured URIs. The `Monitor_Uri_List` property is used for detailed fault monitoring of Sun Java System Web Server. The fault monitor periodically runs the HTTP GET command for the URIs. The monitor takes action if the HTTP request returns with response code 500 "Internal Server Error" or if the application server does not respond. An example URI setting is `http://logical-hostname/App/tester`. If the configured URIs are implemented by using a servlet in the web server, detailed monitoring of the web server Java Virtual Machine (JVM) is possible.

`Probe_timeout`

Type integer. Default is `90`. This property is the time out value (in seconds) that is used by the fault monitor to probe an `iWS` instance. You can modify the value for this property at any time.

Examples

EXAMPLE 1 Creating a Failover iWS Resource in an Existing Group

For this example to work, the data service must first be installed. This example creates a failover `iWS` resource named `webserver-failover` in an existing resource group named `web-rg`. `web-rg` is assumed to contain a `LogicalHostname` resource, which identifies the logical hostname associated with the resource group. Another assumption is that the `Port_list` property defaults to `80/tcp`; that is, the `iWS` instance is listening on port 80.

```
example# clresourcetype register SUNW.iws
example# clresource create -g web-rg -t SUNW.iws \
-p Confdir_list=/global/iws/https-web webserver-failover
```

In this example, the `iws` resource created is named `webserver-failover`, which listens on port 80, with a corresponding `iWS` instance in the directory `/global/iws/https-web`.

EXAMPLE 2 Creating a Scalable iWS Resource

For this example to work, the data service must first be installed. This example creates a scalable `iWS` resource named `webserver-scalable` in a resource group named `web-rg`, which is configured to run simultaneously on all four nodes of a four-node cluster. The `webserver-scalable` resource is configured to listen on port 8080 and uses the IP addresses as configured in a `SharedAddress` resource named `www_foo_com`, which is contained in the resource group `foo_com_RG`.

```
example# clresourcegroup create \
-p Maximum_primaries=4 -p Desired_primaries=4 \
-p RG_dependencies=foo_com_RG web-rg
example# clresourcetype register SUNW.iws
example# clresource create -g web-rg -t SUNW.iws \
-p Confdir_list=/global/iws/https-web \
-p Port_list=8080/tcp -p Scalable=TRUE
\-p Network_resources_used=www_foo_com webserver-scalable
```

EXAMPLE 3 Creating a Failover iWS Resource Listening on a Specified Port

For this example to work, the data service must first be installed. This example creates a failover `iWS` resource named `webserver-secure`, which listens on port 443 in an existing resource group named `web-rg`.

```
example# clresourcetype register SUNW.iws
example# clresource create -g web-rg -t SUNW.iws \
-p Confdir_list=/global/iws/https-web \
-p Port_list=443/tcp webserver-secure
```

In this example, the directory `/global/iws/https-web` must contain a file named `keypass`, which contains the secure key password needed to start the secure web server.

EXAMPLE 4 Creating a Scalable iWS Resource That Contains Two iWS Instances

For this example to work, the data service must first be installed. This example creates a scalable `iWS` resource named `webserver-paired`, which contains two `iWS` instances, one secure and the other non secure. The probe timeout is reduced from the default value of `30` seconds to `20` seconds. The resource listens on the IP addresses contained in two `SharedAddress` resources named `www_foo_com` and `www_foobar_com`. The `Load_balancing_policy` is set to be "sticky" so that a given client always goes to the same cluster node irrespective of whether it contacts the secure port or the non secure one.

```
example# clresourcetype register SUNW.iws
example# clresource create -g web-rg -t SUNW.iws \
-p Confdir_list=/global/iws/https-web-not-secure,/global/iws/https-web-secure \
-p Port_list=80/tcp,443/tcp -p Probe_timeout=20 -y Scalable=TRUE \
-p Network_resources_used=www_foo_com,www_foobar_com
\-p Load_balancing_policy=LB_STICKY webserver-paired
```

EXAMPLE 5 Setting `Monitor_uri_list` for Scalable Sun Java System Web Server Instance

The following example shows how to set the `Monitor_uri_list` extension property when a scalable Sun Java System Web Server instance is added to a configuration. The `Monitor_uri_list` extension property is not supported with a secure Sun Java System Web Server instance.

```
(Add an insecure Sun Java System instance with default load balancing.)

example# clresource create -g resource-group-1 -t SUNW.iws \
-p Confdir_List=/opt/SunONE/https-web-not-secure-1 \
-p Monitor_Uri_list=http://schost-1:8000/servlet/monitor \
-p Scalable=True -y Network_resources_used=schost-1 \
-p Port_list=8000/tcp SunONE-insecure-1
```

EXAMPLE 6 Setting `Monitor_uri_list` for Failover Sun Java System Web Server Instance

The following example shows how to set the `Monitor_uri_list` extension property when a failover Sun Java System Web Server instance is added to a configuration. The `Monitor_uri_list` extension property is not supported with a secure Sun Java System Web Server instance.

```
(Add an insecure Sun Java System application resource instance.)

example# clresource create -g resource-group-1 -t SUNW.iws \
-p Confdir_list=/opt/SunONE/conf \
-p Monitor_Uri_list=http://schost-1:80/servlet/monitor \
-p Scalable=False -y Network_resources_used=schost-1 \
-p Port_list=80/tcp web-not-secure-1
```

Attributes See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWschtt

See Also `clresourcetype`(1CL), `clresourcegroup`(1CL), `scha_resource_get`(1HA), `pmfadm`(1M), `attributes`(5), `r_properties`(5), `scalable_service`(5)

Sun Cluster Data Services Planning and Administration Guide for Solaris OS

Name SUNW.jsas, jsas – resource type implementation for failover and multiple masters Sun Java System Application Server

Description The `SUNW.jsas` resource type represents the Sun Java System Application Server application in a Sun Cluster configuration.

Standard properties and extension properties that are defined for the `SUNW.jsas` resource type are described in the subsections that follow. To set these properties for an instance of the `SUNW.jsas` resource type, use the `clresource`(1CL) commands.

Standard Properties See `r_properties`(5) for a complete description of the following resource properties.

`Network_resources_used`

Default Null string

Tunable When disabled

`Port_list`

Default Null string

Tunable Any time

`Retry_count`

Maximum 10

Default 2

Tunable Any time

`Retry_interval`

Maximum 3600

Default 480

Tunable Any time

`Thorough_probe_interval`

Maximum 3600

Default 120

Tunable Any time

Extension Properties The extension properties of this resource type are as follows:

`Adminuser`
The DAS adminstrative user name.

Data type String array

Default No default defined

Range Not applicable

Tunable At creation

`Confdir_list`
The complete path to the Sun Java System Application Server installation directory.

Data type String array

Default `/opt/SUNWappserver`

Range Not applicable

Tunable At creation

`Domaindir`
The full path to the domain directory.

Data type String

Default Null

Range Not applicable

Tunable At creation

`Domain_name`
The domain name.

Data type String

Default No default defined

Range Not applicable

Tunable At creation

`Monitor_Uri_List`
List of URIs to be probed. The Application Server agent sends HTTP/1.1 GET requests to each of the listed URIs.

The only response code that results in a failover of the resource is the response code 500 (Internal Server Error).

Data type String array

Default Null

Range Not applicable

Tunable Any time

`Passwordfile`
The full path to the file that contains the DAS administrative password.

Data type String

Default No default defined

Range Not applicable

Tunable At creation

`Probe_timeout`
The timeout value (in seconds) for the probe.

Data type Integer

Default 180

Range Minimum = 2

Tunable Any time

Examples

EXAMPLE 1 Creating a Resource for `SUNW.jsas`

For this example to work, you must first install the Sun Cluster HA for Sun Java System Application Server data service. This data service includes all the packages to make Sun Java System Application Server highly available.

This example creates a failover Domain Administration Server (DAS) resource. The resource is created in an existing failover resource group. To create this resource the following commands are run:

```
# clresourcetype register SUNW.jsas
# clresource create -g das-rg -t SUNW.jsas \
-p Adminuser=admin \
-p Domain_name=new-domain \
-p Passwordfile=/global/disk1/passwordfile das-rs
```

In this example, the DAS resource group is named `das-rg`, the DAS resource is named `das-rs`, and values are specified for the extension properties that have no defaults.

Attributes

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWscjsas

See Also

`clresourcetype`(1CL), `clresourcegroup`(1CL), `scha_resource_get`(3HA), `pmfadm`(1M), `attributes`(5), `r_properties`(5), `scalable_service`(5)

Sun Cluster Data Service for Sun Java System Application Server Guide for Solaris OS and *Sun Cluster Data Services Planning and Administration Guide for Solaris OS*

Name SUNW.jsas-na, jsas-na – resource type implementation Sun Java System Application Server

Description The `SUNW.jsas-na` resource type represents the Node Agent component of the Sun Java System Application Server application in a Sun Cluster configuration.

Standard properties and extension properties that are defined for the `SUNW.jsas-na` resource type are described in the subsections that follow. To set these properties for an instance of the `SUNW.jsas-na` resource type, use the `clresource`(1CL) command.

Standard Properties See `r_properties`(5) for a complete description of the following resource properties.

`Network_resources_used`

Default Null string

Tunable When disabled

`Port_list`

Default Null string

Tunable Any time

`Retry_count`

Maximum 10

Default 2

Tunable Any time

`Retry_interval`

Maximum 3600

Default 480

Tunable Any time

`Thorough_probe_interval`

Maximum 3600

Default 120

Tunable Any time

Extension Properties The extension properties of this resource type are as follows:

`Adminhost`
The host name of the Domain Administration Server.

Data type String

Default Null string

Range Not applicable

Tunable When disabled

`Adminport`
The port on which the administration server is listening.

Data type Integer

Default 4849

Range Not applicable

Tunable Any time

`Adminuser`
The Domain Administration Server (DAS) adminstrative user name.

Data type String

Default Null string

Range Not applicable

Tunable When disabled

`Agentdir`
The full path to the Node Agents directory.

Data type String

Default Null string

Range Not applicable

Tunable When disabled

`Confdir_list`
The full path to the Sun Java System Application Server installation directory.

Data type String array

Default `/opt/SUNWappserver`

Range Not applicable

Tunable When disabled

`Passwordfile`
The full path to the file that contains the DAS administrative password and master password.

Data type String

Default Null string

Range Not applicable

Tunable When disabled

`Probe_timeout`
The timeout value (in seconds) for the probe.

Data type Integer

Default 180

Range Minimum = 2

Tunable Any time

Examples

EXAMPLE 1 Creating a Resource for `SUNW.jsas-na`

For this example to work, you must first install the Sun Cluster HA for Sun Java System Application Server data service. This data service includes all the packages to make Sun Java System Application Server highly available.

This example creates a failover Node Agent resource. The resource is created in an existing failover resource group. To create this resource the following commands are run:

```
# clresourcetype register SUNW.jsas-na
# clresource create -g na-rg -t SUNW.jsas-na \
-p Agentdir=/global/disk1/my-domain \
-p Adminuser=admin \
-p Adminhost=host1 \
-p Passwordfile=/global/disk1/passwordfile na-rs
```

In this example, the Node Agent resource group is named `na-rg`, the Node Agent resource is named `na-rs`, and values are specified for the extension properties that have no defaults.

Attributes

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWscjsas

See Also

`clresourcetype`(1CL), `clresourcegroup`(1CL), `scha_resource_get`(3HA), `pmfadm`(1M), `attributes`(5), `r_properties`(5), `scalable_service`(5)

Sun Cluster Data Service for Sun Java System Application Server Guide for Solaris OS and *Sun Cluster Data Services Planning and Administration Guide for Solaris OS*

Name SUNW.krb5, krb5 – resource type implementation of the Kerberos KDC server

Description `SUNW.krb5` is the resource type that uses the `SUNWsckrb5/etc/SUNW.krb5` file to store the resource properties required to drive the high-availability of the Kerberos server.

Standard Properties Standard resource properties are overridden for this resource type as follows:

`Cheap_probe_interval`

Maximum 3600 seconds

Default 30 seconds

Tunable Any time

`Network_resources_used`

Maximum Not applicable

Default No default

Tunable When disabled

`Port_list`

Note – Port 88 is associated with `krb5kdc`(1M) and port 749 belongs to `kadmind`(1M).

Maximum Not applicable

Default 88/tcp, 749/tcp, and 88/udp

Tunable At creation

`Retry_count`

Maximum 10

Default 2

Tunable Any time

`Retry_interval`

Maximum 3600 seconds

Default 600 seconds

Tunable Any time

`Thorough_probe_interval`

Maximum 3600 seconds

Default 300 seconds

Tunable Any time

For more information about standard properties, see the `r_properties`(5) man page.

Extension Properties

The extension properties associated with the `SUNW.krb5` resource type are as follows:

`Monitor_retry_count`
The maximum number of restarts by the process monitor facility (PMF) that are allowed for the fault monitor.

Data type	Integer
Default	4
Range	No range defined
Tunable	Any time

`Monitor_retry_interval`
The period of time in minutes during which the PMF counts restarts of the fault monitor.

Data type	Integer
Default	2 minutes
Range	No range defined
Tunable	Any time

`Probe_timeout`
The time-out value in seconds that the fault monitor uses to probe a Kerberos instance.

Data type	Integer
Default	90 seconds
Range	No range defined
Tunable	Any time

Examples

EXAMPLE 1 Instantiating a Failover Kerberos Resource

This example shows how to instantiate a failover Kerberos resource.

Before you work through this example, ensure that the Sun Cluster HA for Kerberos is installed.

Instantiate a Kerberos resource named `krb5-rs` in a resource group named `krb5-rg`. The `krb5-rg` resource group contains at least one logical hostname resource which identifies the logical hostnames associated with the resource group.

```
# clresourcetype register SUNW.krb5

# clresource create -g krb5-rg -t SUNW.krb5 krb5-rs
```

The resource `krb5-rs` listens on port 88 for `krb5kdc` and 749 for `kadmind`.

Attributes See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWkrb5

See Also `kinit`(1), `kadmin`(1M), `kadmind`(1M), `krb5kdc`(1M), `scrgadm`(1M), `svcadm`(1M), `r_properties`(5), *Sun Cluster Data Service for Kerberos Guide for Solaris OS*

Name SUNW.nfs, nfs – resource type implementation for Sun Cluster HA for NFS

Description The `nfs` resource type implementation operates on a set of `share` commands stored in a "per-resource" file. The format of this file is exactly the same as that described in the `dfstab`(4) man page. This file's location is relative to the `Pathprefix` property of the containing resource group. This file must reside as `SUNW.nfs/dfstab.`*resource_name* under the `Pathprefix` directory that contains the resource group.

Standard Properties See `r_properties`(5) for a complete description of the following resource properties.

`Cheap_probe_interval`

Default `20`

Tunable Any time

`Network_resources_used`

Default `No Default`

Tunable At creation

`Thorough_probe_interval`

Default `60`

Tunable Any time

Extension Properties `Lockd_nullrpc_timeout`
Type integer. Default is `120`. This propery indicates the time out value (in seconds) to use when probing `lockd`.

`Monitor_retry_count`
Type integer. Default is `4`. This property controls the restarts of the fault monitor. It indicates the number of times that the fault monitor is restarted by the Process Monitor Facility (PMF) and corresponds to the `-n` option passed to the `pmfadm`(1M) command. The number of restarts is counted in a specified time window (see the property `Monitor_retry_interval`). Note that this property refers to the restarts of the fault monitor itself, not the NFS daemons.

`Monitor_retry_interval`
Type integer. Default is `2`. This property indicates that the failures of the fault monitor are counted and corresponds to the `-t` option passed to the`pmfadm`(1M) command. If the number of times the fault monitor fails exceeds the extension property `Monitor_retry_count`, the fault monitor is not restarted by the Process Monitor Facility.

`Mountd_nullrpc_restart`
Type Boolean. Default is `TRUE`. Indicates if `mountd` should be restarted when a null `rpc` call fails.

`Mountd_nullrpc_timeout`
Type integer; defaults to `120`. This property indicates the time out value (in seconds) to use when probing `mountd`.

`Nfsd_nullrpc_restart`
Type Boolean. Default is `FALSE`. This property indicates if `nfsd` should be restarted when a null `rpc` call fails.

`Nfsd_nullrpc_timeout`
Type integer. Default is `120`. This property indicates the time out value (in seconds) to use when probing `nfsd`.

`Rpcbind_nullrpc_reboot`
Type Boolean. Default is `TRUE`. Indicates if the system is to be rebooted when a null `rpc` call on `rpcbind` fails.

`Rpcbind_nullrpc_timeout`
Type integer. Default is `120`. This property indicates the time out value (in seconds) to use when probing `rpcbind`.

`Statd_nullrpc_timeout`
Type integer. Defaults to `120`. This property indicates the time out value (in seconds) to use when probing `statd`.

Files

`dfstab.`*resource_name*	The file is in `dfstab` format, which contains the list of share commands to be managed by the resource. This file must reside in the `SUNW.nfs` subdirectory under the `Pathprefix` directory of the containing resource group.
`/tmp/.hanfs/*`	Critical state files used by the implementation.

Examples

EXAMPLE 1 Instantiating a Failover NFS Resource

For this example to work, the data service must first be installed. This example instantiates a failover NFS resource named `hanfs-rs` in a resource group named `hanfs-rg`. The `hanfs-rg` resource group is assumed to contain at least one logical hostname resource, which identifies the logical hostnames associated with the resource group.

```
example# clresourcetype register SUNW.nfs
example# clresource create -g hanfs-rg -t SUNW.nfs hanfs-rs
```

The resource group `hanfs-rg` must contain a valid path name as its `Pathprefix` property. A file named `dfstab.hanfs-rs` must reside in the subdirectory `SUNW.nfs` under the `Pathprefix` directory.

EXAMPLE 2 Instantiating a Failover NFS Resource

For this example to work, the data service must first be installed. This example instantiates a failover NFS resource named `sap-nfs` in a resource group named `sap-rg`. The system-defined property `Thorough_probe_interval` is set to 30 for this resource. The `Network_resources_used` property is set to a logical hostname `relo-sap`, which must reside in the same resource group, `sap-rg`.

```
example# clresourcetype register SUNW.nfs
example# clresource create –g sap-rg –t SUNW.nfs \
-p Thorough_probe_interval=30 \
-p Network_resources_used=relo-sap sap-nfs
```

Attributes

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWnfs

See Also

`lockd`(1M), `mountd`(1M), `nfsd`(1M), `pmfadm`(1M),`rpcbind`(1M), `scha_resource_get`(3HA), `clresourcetype`(1CL), `clresourcegroup`(1CL), `share`(1M), `statd`(1M), `rpc`(3NSL), `dfstab`(4), `attributes`(5), `r_properties`(5)

Sun Cluster Data Services for NFS Guide for Solaris OS, *Sun Cluster Data Services Planning and Administration Guide for Solaris OS*

Notes

The path names being shared by means of `dfstab.`*resource_name* must be unique across all resources, and they cannot be present in the system `dfstab` file on any cluster node.

The implementation supports customization of the `/etc/init.d/nfs.server` script to start the `nfsd` daemon with a customized set of options.

The `SUNW.nfs` subdirectory under the `Pathprefix` directory of the containing resource group is also used by `statd` to save its state.

Name SUNW.oracle_listener, oracle_listener – resource type implementation for the Oracle listener

Description The `SUNW.oracle_listener` resource type represents the Oracle listener in a Sun Cluster configuration. The HA Oracle listener resource is configured with an HA Oracle server resource. For more information, see *Sun Cluster Data Service for Oracle Guide for Solaris OS*.

You must set the following properties for an Oracle listener resource by using `clresource`(1CL).

Standard Properties The standard resource property `Failover` is set for all failover resource types.

See `r_properties`(5) for a complete description of the following resource properties.

`Failover_mode`
Default: `NONE`

Tunable: `Any time`

`Retry_count`
Default: `-1`

Tunable: `Any time`

`Retry_interval`
Default: `600`

Tunable: `Any time`

`Thorough_probe_interval`
Default: `30`

Tunable: `Any time`

Extension Properties `Listener_name`
Type string. Default is `LISTENER`. Defines the name of the listener to be started. This name must match the corresponding entry in the `listener.ora` configuration file. You can change this property only when the resource is disabled.

`Oracle_Home`
Type string. This property is set to the Oracle parent directory that contains the binaries, logs, and parameter files. You can modify this property only when the resource is disabled.

`Probe_timeout`
Type integer. Default is 180. The time-out value in seconds that the fault monitor uses to probe an Oracle listener. You can modify this property at any time.

`User_env`
Type string. Default is `NULL`. This property is set to the name of the file that contains the environment variables to be set before listener startup or shutdown. You can modify this property at any time.

Examples

EXAMPLE 1 Creating a Failover `oracle_listener` Resource

For this example to work, you must first install the data service.

The following example creates a failover `oracle_listener` resource named `ora_listener` in an existing resource group named `oracle-rg`. `oracle-rg` is assumed to contain a `LogicalHostname` resource, which identifies the logical host name associated with the resource group, and an `oracle_server` resource, which identifies the Oracle server associated with the resource group. Another assumption is that the configuration file `listener.ora` was created with the correct port number for the listener to listen at.

```
example# clresourcetype register SUNW.oracle_listener
example# clresource create –g oracle-rg \
–t SUNW.oracle_listener –p ORACLE_HOME=/oracle \
–p LISTENER_NAME=ORALIST ora_listener
```

In this example, the Oracle listener resource created is named `ora_listener`, which has its `ORACLE_HOME` under `/oracle`. The listener name is `ORALIST`, which matches the corresponding entry in the configuration file `listener.ora`.

See Also

`pmfadm`(1M), `scha_resource_get`(1HA), `clresourcetype`(1CL), `clresource`(1CL), `r_properties`(5), SUNW.oracle_server(5)

Sun Cluster Data Service for Oracle Guide for Solaris OS, *Sun Cluster Data Services Planning and Administration Guide for Solaris OS*

Name SUNW.oracle_rac_server, oracle_rac_server – legacy resource type implementation for the Oracle Real Application Clusters (RAC) server managed by Sun Cluster

Description **Note –** This resource type is not required for Sun Cluster 3.2 configurations. This resource type is provided only to ensure that earlier configurations of Sun Cluster Support for Oracle Real Application Clusters continue to function after an upgrade to Sun Cluster 3.2. In Sun Cluster 3.2 configurations, use the SUNW.oracle_rac_server(5) resource type.

The `SUNW.oracle_rac_server` resource type represents the Oracle RAC server in a Sun Cluster 3.2 configuration. Each instance of the Oracle RAC server is represented by a single `SUNW.oracle_rac_server` resource, and is uniquely identified by its `Oracle_Sid` property.

Each resource belongs to a failover resource group. The resource group is restricted to run on only one node. Therefore, the Oracle RAC server resource in the resource group is also restricted to run on only one node. This restriction is enforced by specifying a single node in the node list when the resource group is created.

The resource group manager (RGM) performs only automated startup and shutdown of Oracle RAC server instances. The RGM does not restart or fail over Oracle RAC server instances. The RGM monitors Oracle RAC server resources only to enable the status of these resources to be obtained by using Sun Cluster utilities.

Oracle RAC server instances should be started only after the RAC framework is enabled on the cluster node. To ensure that this requirement is met, configure Oracle RAC server resources and the RAC framework as follows:

- Create a strong positive affinity between the Oracle RAC server resource groups and the RAC framework resource group.
- Create a dependency between the Oracle RAC server resource and the RAC framework resource.

Standard properties and extension properties that are defined for the `SUNW.oracle_rac_server` resource type are described in the subsections that follow. To set these properties for an instance of the `SUNW.oracle_rac_server` resource type, use the `scrgadm`(1M) command.

Standard Properties The standard resource property `Failover` is set for all failover resource types.

Standard resource properties are overridden for this resource type as follows:

`Failover_mode`	Default: `NONE` Tunable: `Any time`.
`Thorough_probe_interval`	Default: 30 Tunable: `Any time`.

For a description of standard resource properties, see the `r_properties`(5) man page.

Extension Properties `Auto_End_Bkp`

Type Boolean; defaults to `False`. This property specifies whether the Oracle RAC server resource automatically recovers the database if an Oracle relational database management system (RDBMS) hot backup is interrupted. When a hot backup is interrupted, the database fails to open because of files that remain in hot backup mode. During the startup of the Oracle RAC server resource, the resource tests for the interruption of a hot backup by testing for an occurrence the following RDBMS error:

`ORA-01113 file` *file* `needs media recovery`

To recover the database automatically, the Oracle RAC server resource performs the following actions:

- Releasing all files that remain in hot backup mode. The `sys.v$backup` view indicates which files remain in hot backup mode.
- Opening the database for use.

The permitted values for this property are as follows:

`False` Specifies that the Oracle RAC server resource does *not* automatically recover the database. If a hot backup is interrupted, you must recover the database manually. In this situation, the status of the Oracle RAC server resource is set to `FAULTED`. The default value of this property is `False`.

`True` Specifies that the Oracle RAC server resource automatically recovers the database.

You can modify this property at any time.

`Debug_level`

Type integer; defaults to 1, which logs `syslog` messages. This property indicates the level to which debug messages from the Oracle RAC server component are logged. When the debug level is increased, more debug messages are written to the log files. You can modify this property at any time.

`Oracle_Home`

Type string. This property is set to the path to the Oracle home directory. The Oracle home directory contains the binary files, log files, and parameter files for the Oracle software. You can modify this property only when the resource is disabled.

`Oracle_Sid`

Type string. This property is set to the Oracle system identifier. This identifier is the name of the Oracle database instance. You can modify this property only when the resource is disabled.

`Parameter_file`

Type string. This property is set to the Oracle parameter file, which starts the database. If this property is not set, it defaults to `NULL`. When this property is `NULL`, the default Oracle mechanism is used to locate the parameter file. You can modify this property at any time.

`User_env`
Type string; defaults to `NULL`. This property is set to the name of the file that contains the environment variables to be set before database startup or shutdown. All environment variables that have values that differ from Oracle defaults must be defined in this file.

For example, a user's `listener.ora` file might not reside under the `/var/opt/oracle` directory or the `$ORACLE_HOME/network/admin` directory. In this situation, the `TNS_ADMIN` environment variable should be defined.

The definition of each environment variable that is defined must follow the format *variable-name*=*value*. Each definition must start on a new line in the environment file.

You can modify this property at any time.

`Wait_for_online`
Type Boolean; defaults to `True`. This property specifies whether the `START` method of the Oracle RAC server resource waits for the database to be online before the `START` method exits. The permitted values for this property are as follows:

`True` Specifies that the `START` method of the Oracle RAC server resource waits for the database to be online before the `START` method exits. The default value of this property is `True`.

`False` Specifies that the `START` method runs the commands to start the database but does not wait for the database to come online before the `START` method exits.

You can modify this property at any time.

Examples

EXAMPLE 1 Creating `oracle_rac_server` Resources and Resource Groups

The following example creates two `oracle_rac_server` resources and two `oracle_rac_server` resource groups on a two-node cluster. One `oracle_rac_server` resource group is created for each `oracle_rac_server` resource. Each resource and its corresponding resource group are named as shown in the following table.

Resource Name	Resource Group Name
`RAC1-rs`	`RAC1-rg`
`RAC2-rs`	`RAC2-rg`

The example assumes that a `SUNW.rac_framework` resource group named `rac-framework-rg` has been created. The example also assumes that the constituent resources of this resource group have been created. These resources are instances of resource types as shown in the following table.

Resource Type	Resource Name
SUNW.rac_framework	rac_framework
SUNW.rac_udlm	rac_udlm
SUNW.rac_svm	rac_svm

```
example# scrgadm -a -t SUNW.oracle_rac_server
example# scrgadm -a -g RAC1-rg -h node1 \
         -y RG_AFFINITIES=++rac-framework-rg
example# scrgadm -a -g RAC2-rg -h node2 \
         -y RG_AFFINITIES=++rac-framework-rg
example# scrgadm -a -j RAC1-rs -g RAC1-rg \
         -t SUNW.oracle_rac_server \
         -y RESOURCE_DEPENDENCIES=rac_framework \
         -x ORACLE_SID=RAC1 \
         -x ORACLE_HOME=/oracle
example# scrgadm -a -j RAC2-rs -g  RAC2-rg \
         -t SUNW.oracle_rac_server \
         -y RESOURCE_DEPENDENCIES=rac_framework \
         -x ORACLE_SID=RAC2 \
         -x ORACLE_HOME=/oracle
```

The preceding commands create two `oracle_rac_server` resources and two `oracle_rac_server` resource groups on a two-node cluster by performing the following operations:

1. Registering the `SUNW.oracle_rac_server` resource type
2. Creating the `RAC1-rg` resource group for node `node1`
3. Creating the `RAC2-rg` resource group for node `node2`
4. Creating the `RAC1-rs` resource in the `RAC1-rg` resource group for node `node1`
5. Creating the `RAC2-rs` resource in the `RAC2-rg` resource group for node `node2`

Attributes

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWscor

See Also

`scrgadm`(1M), `scswitch`(1M), `attributes`(5), SUNW.oracle_listener(5), `SUNW.rac_cvm`(5), `SUNW.rac_framework`(5), `SUNW.rac_svm`(5), `SUNW.rac_udlm`(5), SUNW.scalable_rac_server(5)

Sun Cluster Data Service for Oracle RAC Guide for Solaris OS, *Sun Cluster Data Services Planning and Administration Guide for Solaris OS*

Name SUNW.oracle_server, oracle_server – resource type implementation for HA Oracle server

Description The `SUNW.oracle_server` resource type represents the HA Oracle server in a Sun Cluster configuration. The HA Oracle server resource is configured with an Oracle listener resource. For more information, see *Sun Cluster Data Service for Oracle Guide for Solaris OS*.

You must set the following properties for an Oracle server resource by using `clresource`(1CL).

Standard Properties The standard resource property `Failover` is set for all failover resource types.

See `r_properties`(5) for a complete description of the following resource properties.

`Failover_mode`
Default: `SOFT`

Tunable: `Any time`

`Retry_count`
Default: `2`

Tunable: `Any time`

`Retry_interval`
Default: `1330`

Tunable: `Any time`

`Thorough_probe_interval`
Default: `30`

Tunable: `Any time`

Extension Properties `Alert_log_file`
Type string. This property is set to the absolute path of the Oracle alert log file. The Oracle software logs alerts in this file. The Oracle server fault monitor scans the alert log file for new alerts at the following times:

- When the server fault monitor is started
- Each time that the server fault monitor queries the health of the server

If an action is defined for a logged alert that the server fault monitor detects, the server fault monitor performs the action in response to the alert.

Preset actions for logged alerts are listed in Appendix B, "Preset Actions for DBMS Errors and Logged Alerts," in *Sun Cluster Data Service for Oracle Guide for Solaris OS* in *Sun Cluster Data Service for Oracle Guide for Solaris OS*. To change the action that the server fault monitor performs, customize the server fault monitor as explained in "Customizing the Sun Cluster HA for Oracle Server Fault Monitor" in *Sun Cluster Data Service for Oracle Guide for Solaris OS* in *Sun Cluster Data Service for Oracle Guide for Solaris OS*.

You can modify this parameter any time.

`Auto_end_bkp`
Type Boolean. Default is `FALSE`. This property instructs the `oracle_server` `START` method to automatically recover the database during startup if the database had previously been interrupted during a hot backup.

If this property is set to `TRUE`, and the `oracle_server` `START` method detects the RDBMS error `ORA-01113 file %s needs media recovery` during startup, all files left in hot backup mode (as indicated by the `sys.v$backup` view). These files are automatically taken out of hot backup mode by using the command:

`alter database datafile` '*filename*' **`end backup;`**

The database is then opened for use.

If this property is set to `FALSE`, the `oracle_server` `START` method takes no recovery action following an `ORA-01113` error, and the status of the resource is set to `FAULTED`. Manual intervention is required at this stage.

This property can be modified at any time.

`Connect_cycle`
Type integer. Default is 5. The Oracle server fault monitor connects to the database periodically by using the user ID and password specified in `Connect_string`. The monitor disconnects after executing the number of probes specified in this property and then reconnects. You can modify this property at any time.

`Connect_string`
Type string. This property is set to the user ID and password of the database user in fault-monitor transactions. This property is specified as follows:

userid/*password*

As part of the HA Oracle setup, you must define the database user ID and password before enabling the server resource and its fault monitor. To use Solaris authentication, type a slash (`/`) instead of a user ID and password. This property must be set for standby databases as well. This property is used by the fault monitor after the physical standby database is transitioned to a primary database. You can modify this property at any time.

`Custom_action_file`
Type string. Default is `NULL`. Introduced in release 3.1 10/03. This property specifies the absolute path of the file that defines the custom behavior of the Sun Cluster HA for Oracle server fault monitor. The format of this file is defined in the custom_action_file(4) man page. You can modify this property at any time.

`Dataguard_role`
Type string. This property specifies the role of the database. The permitted values for this property are as follows:

`NONE`
Specifies that no standby database instances are configured for the primary database instance.

`PRIMARY`
Specifies that the database is a primary database instance for which standby database instances are configured.

`STANDBY`
Specifies that the database role is standby. This value is used by Sun Cluster Ha for Oracle data service along with the `Standby_mode` property value to determine the role of the database.

`IN_TRANSITION`
Specifies that the database is undergoing a role reversal process. This value must be set, when a role reversal process is to be applied to the database. The `Dataguard_role` and `Standby_mode` properties must be set after the role reversal process is complete to reflect the correct role of the database.

You can modify this property at any time.

`Debug_level`
Type integer. Default is 1. This property indicates the level to which debug messages from the fault monitor of the Oracle server component are logged. When the debug level is increased, more debug messages are written to the log files. These messages are logged to the file `/var/opt/SUNWscor/oracle_server/message_log.`*rs*, where *rs* is the name of the resource that represents the Oracle server component. You can modify this property at any time.

`Oracle_Home`
Type string. This property is set to the Oracle parent directory that contains the binaries, logs, and parameter files. You can modify this property only when the resource is disabled.

`Oracle_Sid`
Type string. This property is set to the name of the Oracle database instance (also called the Oracle system identifier). You can modify this property only when the resource is disabled.

`Parameter_file`
Type string. This property is set to the Oracle parameter file, which starts the database. If this property is not set, it defaults to `$ORACLE_HOME/dbs/init$ORACLE_SID.ora`. If the default value is not found, Sun Cluster HA for Oracle checks for `$ORACLE_HOME/dbs/spfile$ORACLE_SID.ora`. You can modify this property at any time.

`Probe_timeout`
Type integer. Default is 300 seconds. This property is the timeout value (in seconds) that is used by the fault monitor to probe an Oracle server instance. You can modify this property at any time.

`Restart_type`
Type enumeration. Default is `RESOURCE_RESTART`. This property specifies the entity that the server fault monitor restarts when the response to a fault is restart. The permitted values for this property are as follows:

`RESOURCE_RESTART`	Specifies that only this resource is restarted.
`RESOURCE_GROUP_RESTART`	Specifies that all resources in the resource group that contains this resource are restarted.

You can modify this property at any time.

`Standby_mode`
Type string. Default is `LOGICAL`. This property specifies the mode of the standby database. This property is used by Sun Cluster HA for Oracle data service only when the `Dataguard_role` property is set to `STANDBY` to determine the type of standby database. The permitted values for this property are as follows:

`LOGICAL`	Specifies a logical standby database.
`PHYSICAL`	Specifies a physical standby database.

You can modify this property at any time.

`User_env`
Type string. Default is `NULL`. This property is set to the name of the file that contains the environment variables to be set before database startup or shutdown. You can modify this property at any time.

`Wait_for_online`
Type Boolean. Default is `TRUE`. This property specifies whether the `oracle_server` `START` method waits for the database to be online before exiting. If this property is set to `FALSE`, `oracle_server` `START` executes the commands to start the database but does not wait for it to come online before exiting. You can modify this property at any time.

Examples

EXAMPLE 1 Creating a Failover `oracle_server` Resource

For this example to work, the data service must forst be installed.

The following example creates a failover `oracle_server` resource named `ora_server` in an existing resource group named `oracle-rg`. `oracle-rg` is assumed to contain a `LogicalHostname` resource, which identifies the logical hostname associated with the resource group, and an `oracle_listener` resource, which identifies the Oracle listener associated with the resource group.

```
example# clresourcetype register SUNW.oracle_server
example# clresource create –g oracle-rg \
–t SUNW.oracle_server -p CONNECT_STRING=scott/tiger \
–p ORACLE_SID=oraSID –p ORACLE_HOME=/oracle \
```

EXAMPLE 1 Creating a Failover `oracle_server` Resource *(Continued)*

```
–p ALERT_LOG_FILE=/oracle/admin/oraSID/bdump/alert_oraSID.log ora_server
```

In this example, the Oracle server resource created is named `ora_server`, which has its `ORACLE_HOME` under `/oracle`. The SID of the is `ora_server` is `oraSID`. Its fault monitor uses the user ID `scott` and the password `tiger` to connect to the database. The alert log file scanned by the fault monitor for any errors that have occurred is at `/oracle/admin/oraSID/bdump/alert_oraSID.log`.

See Also `pmfadm`(1M), `scha_resource_get`(1HA), `clresourcetype`(1CL), `clresource`(1CL), custom_action_file(4), SUNW.oracle_listener(5)

Sun Cluster Data Services Planning and Administration Guide for Solaris OS, *Sun Cluster Data Service for Oracle Guide for Solaris OS*

Name SUNW.s1as, s1as – resource type implementation for failover and multiple masters Sun Java System Application Server (s1as)

Description The `SUNW.s1as` resource type represents the Sun Java System Application Server application in a Sun Cluster configuration.

You must set the following properties on an `SUNW.s1as` resource by using `scrgadm`(1M).

Standard Properties See `r_properties`(5) for a complete description of the following resource properties.

`Network_resources_used`	Default: `No default` Tunable: `At creation`
`Port_list`	Default: `No default` Tunable: `Any time`
`Retry_count`	Default: `2` Tunable: `Any time`
`Retry_interval`	Default: `480` Tunable: `Any time`
`Thorough_probe_interval`	Default: `120` Tunable: `Any time`

Extension Properties `Confdir_list`
Type string array. This property is a path name set to *install_dir*/domains/*domain*/*server* which is the path name of an `s1as` instance directory. You must specify this property at resource creation time.

`Monitor_Uri_List`
Type string array; defaults to "". This property allows you to ensure that application components are responding by querying the configured URIs. The `Monitor_Uri_List` property is used for detailed fault monitoring of Sun Java System Application Server. The fault monitor periodically performs HTTP GET command for the URIs. The monitor takes action if the HTTP request returns with response code 500 "Internal Server Error" or if the application server does not respond. An example URI setting is `http://logical-hostname/App/tester`. If the configured URIs are implemented by using a Servlet in the application server, detailed monitoring of the application server JVM (Java Virtual Machine) is possible. Either the `Monitor_Uri_List` or the `Network_Resources_Used` and the `Port_List` properties must be set. If `Network_Resources_Used`, `Port_List`, and `Monitor_Uri_List` are all set, the fault monitor will probe the ports and the URIs provided. Setting `Port_List` to include the IIOP listeners helps to ensure that the application server is listening and responding to IIOP requests.

`Probe_timeout`
Type string; defaults to 120 seconds. This property is tunable anytime and sets the timeout value for the probe.

Examples

EXAMPLE 1 Creating a Failover s1as Resource in an Existing Group

For this example to work, you must first install the `SUNWscs1as` data service package. This example creates a failover `s1as` resource named `appsrv-rs` in an existing resource group named `appsrv-rg`. The `appsrv-rg` resource group contains a `LogicalHostname` resource, which identifies the logical hostname associated with the resource group.

```
(Register the SUNW.s1as resource type.)
# scrgadm -a -t SUNW.s1as

(Create a Sun Java System Application Server resource and add it to the
resource group.)
# scrgadm -a -j appsrv-rs -g appsrv-rg \\
-t SUNW.s1as \\
-x Confdir_list=/global/appsrv/domains/scdomain/server1 \\
-y Network_resources_used=schost-1 \\
-y Port_list=80/tcp,3700/tcp \\
-x Monitor_Uri_list=http://schost-1:80/servlet/monitor
```

In the preceding example, the `s1as` resource created is named `appserver-rs`, with a corresponding `s1as` instance in the directory `/global/appsrv/domains/scdomain/server1`.

EXAMPLE 2 Creating a Failover s1as Resource Listening on a Specified Port

For this example to work, you must first install the `SUNWscs1as` data service package. This example creates a failover `s1as` resource named `appserver-secure`, which listens on port 443 in an existing resource group named `app-rg`.

```
example# scrgadm -a -t SUNW.s1as
example# scrgadm -a -j appserver-secure -t SUNW.s1as
 -g app-rg
-x Confdir_list=/global/s1as/domains/domain1/server2
-y Network_resources_used=schost-1
-x Monitor_Uri_list=http://schost-1:80/servlet/monitor
-y Port_list=443/tcp
```

EXAMPLE 3 Configuring s1as in a Scalable Resource Group

In this example a scalable resource group, `s1as-rg`, is created with `Maximum_primaries` and `Desired_primaries` equal to three. A Sun Java System Application Server resource named `scalable-app-server1` is then created in this resource group with a configuration directory of `/global/s1as/domains/domain1/server1` . The URI `http://localhost:8000/servlets/testservlet` is specified to the extension property `Monitor_uri_list`.

EXAMPLE 3 Configuring s1as in a Scalable Resource Group *(Continued)*

```
example# scrgadm -a -g s1as-rg
-y Maximum_primaries=3 -y Desired_primaries=3
example# scrgadm -a -g s1as-rg -j scalable-app-server1 -t SUNW.s1as
-x Confdir_list=/global/s1as/domains/domain1/server1
-x Monitor_uri_list=http://localhost:8000/servlets/testservlet
```

Attributes

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWscs1as

See Also

`scrgadm`(1M), `scswitch`(1M), `scha_resource_get`(3HA), `pmfadm`(1M), `attributes`(5), `r_properties`(5), `scalable_service`(5)

Sun Cluster Data Service for Sun Java System Application Server Guide for Solaris OS, and *Sun Cluster Data Services Planning and Administration Guide for Solaris OS*

Name SUNW.s1mq, s1mq – resource type implementation for failover Sun Java System Message Queue (s1mq)

Description The `SUNW.s1mq` resource type represents the Sun Java System Message Queue application in a Sun Cluster configuration.

You must set the following properties on an `SUNW.s1mq` resource by using `clresource`(1CL).

Standard Properties See `r_properties`(5) for a complete description of the following resource properties.

`Network_resources_used`

Default `No Default`

Tunable At creation

If set, the `Network_resources_used` property is used to determine which network resources the fault monitor will use to probe the message broker. If the `Network_resources_used` property is not set, the fault monitor will probe the message broker on all of the network resources contained in the resource group that contains the Message Queue resource.

`Port_list`

Default `No default`

Tunable At creation

`Retry_count`

Default `2`

Tunable Any time

`Retry_interval`

Default `300`

Default Any time

`Thorough_probe_interval`

Default `60`

Tunable Any time

Extension Properties `Confdir_list`

Type string array. This property is a path name set to *install_dir/domains/domain/server* which is the path name of an `s1mq` instance directory. You must specify this property at resource creation time.

`Broker_Name`
Type string. No default. This property contains the name of the broker to start and monitor. The `imqcmd` command needs this name to stop the broker if `Smooth_Shutdown` is set to `TRUE`.

`Broker_User`
Type string. Default is "". This property contains the Message Queue user name of the managed broker. This user name is used to shut down the broker if `Smooth_Shutdown` is set to `TRUE`. `Smooth_Shutdown` defaults to `FALSE`. If `Smooth_Shutdown=FALSE`, the broker is sent `SIGTERM` to shut it down. If `Smooth_Shutdown` is set to `TRUE` the broker will be shut down by using `imqcmd`. Using `imqcmd` exposes the broker user password on the `imqcmd` command line.

`Probe_timeout`
Type string. Default is 180 seconds. This property is tunable at anytime and sets the timeout value for the probe.

Examples

EXAMPLE 1 Creating a Failover `s1mq` Resource in an Existing Group

For this example to work, the `SUNWscs1mq` data service package must first be installed. This example creates a failover `s1mq` resource named `message-queue-failover` in an existing resource group named `mq-rg`. The `mq-rg` resource group contains a `LogicalHostname` resource, which identifies the logical hostname associated with the resource group.

```
example# clresourcetype register SUNW.s1mq
example# clresource create -g mq-rg -t SUNW.s1mq \
-p Confdir_list=/global/s1mq/instances/hamq1 \
-p Network_Resources_used=logical host \
-p Port_List=7676\\tcp \
-p Broker_Name=hamq1 message-queue-failover
```

In the preceding example, the `s1mq` resource created is named `message-queue-failover`. The s1mq resource listens on port 7676, with a corresponding `s1mq` instance in the directory `/global/s1mq/instances/hamq1`.

EXAMPLE 2 Creating a Failover `s1mq` Resource with `Smooth_Shutdown=TRUE`

For this example to work, the `SUNWscs1mq` data service package must be first installed. This example creates a failover `s1mq` resource named `message-queue-failover`, which listens on port 7676 in an existing resource group named `mq-rg`.

```
example# clresourcetype register SUNW.s1mq
example# clresource create -g mq-rg -t SUNW.s1mq \
-p Confdir_list=/global/s1mq/instances/hamq1 \
-p Network_Resources_used=logical host \
-p Port_List=7676 -p Broker_Name=hamq1 -p Broker_User=admin \
-p Smooth-Shutdown=TRUE message-queue-failover
```

Attributes See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWscs1mq

See Also `clresourcetype`(1CL), `clresourcegroup`(1CL), `scha_resource_get`(3HA), `pmfadm`(1M), `attributes`(5), `r_properties`(5), `scalable_service`(5)

Sun Cluster Data Service for Sun Java System Message Queue Guide for Solaris OS, and *Sun Cluster Data Services Planning and Administration Guide for Solaris OS*

Name SUNW.sap_as, sap_as, SUNW.sap_as_v2, sap_as_v2 – resource type implementations for Sun Cluster HA for SAP application server

Description The `sap_as` or `SUNW.sap_as` resource type represents Sun Cluster HA for SAP as a failover data service.

The `sap_as_v2` or `SUNW.sap_as_v2` resource type represents Sun Cluster HA for SAP as a failover data service or a scalable data service.

The Resource Group Manager (RGM) manages the SAP data service for Sun Cluster software. If you are setting up the Sun Cluster HA for SAP application server as a failover data service, configure it as a logical hostname resource and an SAP application server resource. If you are setting up the *Sun Cluster HA for SAP application server*Sun Cluster HA for SAP application server as a scalable data service, configure it as a scalable SAP application server resource.

Use the `clresource`(1CL) command or a resource configuration GUI to set the following properties on an SAP application server resource.

Standard Properties See `r_properties`(5) for a complete description of the following resource properties.

`Failover_mode`	Default: `SOFT` Tunable: `Any time`
`Retry_count`	Default: `3` Tunable: `Any time`
`Retry_interval`	`Default: 3600` Tunable: `Any time`
`Thorough_probe_interval`	Default: `60` Tunable: `Any time`

Extension Properties `SAPSID`
Type string. No default exists for this field. You must provide the value when you create the resource. The value is the SAP system name or *SAPSID*. You can modify this property only when you have disabled the resource.

`As_instance_id`
Type string. This value is a two-digit SAP system number or instance ID. No default exists for this field. You must provide the value when you create the resource. You can modify this property only when you have disabled the resource.

`As_services_string`
Type string. The default is `D`, which is a string of services that the application server provides. You can modify this property only when you have disabled the resource.

`Monitor_retry_count`
Type integer. The default is `4`. This property controls fault-monitor restarts. The property indicates the number of times the process monitor facility (PMF) restarts the fault monitor. The property corresponds to the `-n` option passed to the `pmfadm`(1M) command. The RGM counts the number of restarts in a specified time window (see the property `Monitor_retry_interval`). Note that this property refers to the restarts of the fault monitor itself, not the SAP application server. You can modify the value for this property at any time.

`Monitor_retry_interval`
Type integer. The default is `2`. This property indicates the time window in minutes during which the RGM counts fault-monitor failures. The property corresponds to the `-t` option passed to the `pmfadm`(1M) command. If the number of times the fault monitor fails exceeds the extension property `Monitor_retry_count`, the PMF does not restart the fault monitor. You can modify the value for this property at any time.

`As_db_retry_interval`
Type integer. The default is `30`. This property indicates the time window in seconds to wait between attempts to connect to the database before starting the SAP application server at startup time, if the database is unavailable. You can modify this property only when you have disabled the resource.

`As_startup_script`
Type string. The Sun Cluster HA for SAP data service uses the startup script name to start the SAP application server. No default exists for this field. You must supply the value when you create the resource. You can modify this property only when you have disabled the resource.

`Stop_sap_pct`
Type integer. The default is `95`. This property indicates the percentage of the `Stop_timeout` value that the Sun Cluster HA for SAP data service uses to stop SAP processes with the SAP shutdown script before calling PMF to terminate the SAP processes. You can modify this property only when you have disabled the resource.

`As_shutdown_script`
Type string. This value is the shutdown script name, which the Sun Cluster HA for SAP data service uses to shut down the SAP application server. No default exists for this field. You must supply the value when you create the resource. You can modify this property only when you have disabled the resource.

`Probe_timeout`
The default is `120`. This property indicates the timeout value in seconds for the probe. You can modify the value for this property at any time.

Attributes See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWscsap

See Also `pmfadm`(1M), `scha_resource_get`(1HA), `clresourcetype`(1CL), `clresource`(1CL), `attributes`(5), SUNW.sap_ci(5)

Sun Cluster Data Services Planning and Administration Guide for Solaris OS

Name SUNW.sap_ci, sap_ci, SUNW.sap_ci_v2, sap_ci_v2 – resource type implementations for Sun Cluster HA for SAP central instance.

Description The Resource Group Manager (RGM) manages the SAP data service for Sun Cluster software. Configure the Sun Cluster HA for SAP central instance as a logical hostname resource and an SAP central instance resource.

Use the `clresourcetype`(1CL) command or a resource configuration GUI to set the following properties on an SAP central-instance resource.

Standard Properties See `r_properties`(5) for a complete description of the following resource properties.

`Failover_mode`	Default: `SOFT` Tunable: Any time
`Retry_count`	Default: `3` Tunable: `Any time`
`Retry_interval`	Default: `3600` Tunable: `Any time`
`Thorough_probe_interval`	Default: `60` Tunable: `Any time`

Extension Properties `SAPSID`
Type string. No default exists for this field. You must provide the value when you create the resource. The value is the SAP system name or *SAPSID*. You can modify this property only when you have disabled the resource.

`Ci_instance_id`
Type string. The default is `00`. This value is a two-digit SAP system number or instance ID. This value is a two-digit SAP system number or instance ID. You can modify this property only when you have disabled the resource.

`Ci_services_string`
Type string. The default is `DVEBMGS`, which is a string of services that the central instance provides. You can modify this property only when you have disabled the resource.

`Message_server_name`
Type string. The default is `sapms`*SAPSID*. This value is the name of the message server. You can modify this property only when you have disabled the resource.

`Monitor_retry_count`
Type integer. The default is `4`. This property controls fault-monitor restarts. The property indicates the number of times the process monitor facility (PMF) restarts the fault monitor. The property corresponds to the `-n` option passed to the `pmfadm`(1M) command. The RGM

counts the number of restarts in a specified time window (see the property `Monitor_retry_interval`). Note that this property refers to the restarts of the fault monitor itself, not the SAP central instance. You can modify the value for this property at any time.

`Monitor_retry_interval`
Type integer. The default is `2`. This property indicates the time window in minutes during which the RGM counts fault-monitor failures. The property corresponds to the `-t` option passed to the `pmfadm`(1M) command. If the number of times the fault monitor fails exceeds the extension property `Monitor_retry_count`, the PMF does not restart the fault monitor. You can modify the value for this property at any time.

`Probe_timeout`
The default is 120. This property indicates the time-out value in seconds for the probes. You can modify the value for this property at any time.

`Check_ms_retry`
Type integer. The default is `2`. This property indicates the maximum number of times the SAP message-server check can fail before the fault monitor reports a total failure. You can modify this property only when you have disabled the resource.

`Ci_start_retry_interval`
Type integer. The default is `30`. This property indicates the time window in seconds to wait between attempts to connect to the database before starting the SAP central instance at startup time, if the database is unavailable. You can modify this property only when you have disabled the resource.

`Ci_startup_script`
Type string. The Sun Cluster HA for SAP data service uses the startup script name to start the SAP central instance. No default value exists. You must supply the value when you create the resource. You can modify this property only when you have disabled the resource.

`Stop_sap_pct`
Type integer. The default is `95`. This property indicates the percentage of the `Stop_timeout` value that the Sun Cluster HA for SAP data service uses to stop SAP processes with the SAP shutdown script before calling PMF to terminate the SAP processes. You can modify this property only when you have disabled the resource.

`Ci_shutdown_script`
Type string. This value is the shutdown script name, which the Sun Cluster HA for SAP data service uses to shut down the SAP central instance. No default value exists. You must supply the value when you create the resource. You can modify this property only when you have disabled the resource.

`Lgtst_ms_with_logicalhostname`
Type boolean. The default is `TRUE`. This property indicates how to check the SAP message server with the SAP `lgtst` utility. The `lgtst` utility requires a hostname (IP address) as the location for the SAP message server. This hostname can be either a Sun Cluster logical

hostname or a `localhost` (loop back) name. If you set this resource property to `TRUE`, use a logical hostname. Otherwise, use a local hostname. This property indicates whether the fault monitor probes the message server using the utility `lgtst` with the logical hostname. If you change this value to `FALSE`, the fault monitor uses the loopback address instead of the logical hostname with the utility `lgtst`. You can modify the value for this property at any time.

`Shutdown_dev`
Type boolean. The default is `FALSE`. This property indicates whether the RGM shuts down a development system before starting the SAP central instance. If you set this value to `TRUE`, you must also set the extension properties `Dev_sapsid` and `Dev_shutdown_script`. You can modify this property only when you have disabled the resource.

`Dev_sapsid`
Type string. This value is the development system name or *SAPSID*. You must set this property if you set the property `Shutdown_dev` to the value `TRUE`. You can modify this property only when you have disabled the resource.

`Dev_shutdown_script`
Type string. This value is the name of the shutdown script, which the RGM uses to shut down the SAP development system. No default value exists. You must supply the value when you create the resource if you set the property `Shutdown_dev` to the value `TRUE`. You can modify this property only when you have disabled the resource.

`Dev_stop_pct`
Type integer. The default is `20`. This property indicates the percentage of the `Start_timeout` value that the Sun Cluster HA for SAP data service uses to shut down the development system before starting the SAP central instance. The Sun Cluster HA for SAP data service will not use this property if you set the property `Shutdown_dev` to the value `FALSE`. You can modify this property only when you have disabled the resource.

Attributes

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWscsap

See Also

`pmfadm`(1M), `scha_resource_get`(1HA), `clresourcegroup`(1CL)`clresourcetype`(1CL),`clresource`(1CL),`attributes`(5), `r_properties`(5), SUNW.sap_as(5)

Sun Cluster Data Services Planning and Administration Guide for Solaris OS

Name SUNW.sapdb, sapdb – resource type implementation for Sun Cluster HA for MaxDB

Description The `SUNW.sapdb` resource type represents the MaxDB application in a Sun Cluster configuration. The MaxDB application requires the SAP xserver system. Therefore, you must set a dependency between the MaxDB resource group and the SAP xserver resource group. Create this dependency when you register and configure the Sun Cluster HA for MaxDB data service. For more information, see "Registering and Configuring Sun Cluster HA for MaxDB" in *Sun Cluster Data Service for MaxDB Guide for Solaris OS* in *Sun Cluster 3.2 Data Service for MaxDB Guide*.

Standard properties and extension properties that are defined for the `SUNW.sapdb` resource type are described in the subsections that follow. To set these properties for an instance of the `SUNW.sapdb` resource type, use the `clresourcetype`(1CL) command.

Standard Properties Standard resource properties are overridden for this resource type as follows:

`Retry_Count`

Maximum 10

Default 2

Tunable Any time

`Retry_Interval`

Maximum 3600

Default 850

Tunable Any time

`Thorough_Probe_Interval`

Maximum 3600

Default 120

Tunable Any time

For a description of these standard resource properties, see `r_properties`(5).

Extension Properties The extension properties of this resource type are as follows:

`dbmcli_Start_Option`
The option that is passed to the `dbmcli` command to start the MaxDB database instance.

Note – For MaxDB version 7.4.3, set this property to `db_online`.

Data type String

Default `db_online`

Range Not applicable

Tunable When disabled

`DB_Name`
The name of the MaxDB database instance in uppercase. This name is created when MaxDB is installed and configured as explained in "Installing and Configuring MaxDB" in *Sun Cluster Data Service for MaxDB Guide for Solaris OS* in *Sun Cluster 3.2 Data Service for MaxDB Guide*.

Data type String

Default No default defined

Range Not applicable

Tunable When disabled

`DB_User`
The UNIX user identity of the operating system (OS) user that administers the MaxDB database instance. This user's home directory contains the `.XUSER.62` file that was created during the installation and configuration of MaxDB. For more information, see "Installing and Configuring MaxDB" in *Sun Cluster Data Service for MaxDB Guide for Solaris OS* in *Sun Cluster 3.2 Data Service for MaxDB Guide*.

Data type String

Default No default defined

Range Not applicable

Tunable When disabled

`Failover_enabled`
Specifies whether the fault monitor fails over the MaxDB resource if the number of attempts to restart exceeds `Retry_count` within the time that `Retry_interval` specifies. The possible values of this extension property are as follows:

- `True` – Specifies that the fault monitor fails over the MaxDB resource
- `False` – Specifies that the fault monitor does *not* fail over the MaxDB resource

Data type Boolean

Default `True`

Range Not applicable

Tunable Any time

`Independent_Program_Path`
The full path to the directory that contains the following programs and libraries for the MaxDB application:

- Programs that are independent of the database software version
- Libraries for the client runtime environment

Sun Cluster HA for MaxDB determines the path to the `dbmcli` command from the value of this property. The `dbmcli` command resides in the `bin` subdirectory of the directory that this property specifies.

Data type String

Default `/sapdb/programs`

Range Not applicable

Tunable When disabled

`Monitor_retry_count`
The maximum number of restarts by the process monitor facility (PMF) that are allowed for the fault monitor.

Data type Integer

Default 4

Range No range defined

Tunable Any time

`Monitor_retry_interval`
The period of time in minutes during which the PMF counts restarts of the fault monitor.

Data type Integer

Default 2

Range No range defined

Tunable Any time

`Pid_Dir_Path`
The full path to the directory under which files that store the process identities of MaxDB kernel processes are created. The process identities of MaxDB kernel processes are stored in the following files:

- *pid-dir*`/ppid/`*db-name*
- *pid-dir*`/pid/`*db-name*

The replaceable items in these file paths are as follows:

- *pid-dir* is the directory that the `Pid_Dir_Path` extension property specifies
- *db-name* is the name of the MaxDB database instance that the `DB_Name` extension property specifies

Data type String

Default `/var/spool/sql`

Range Not applicable

Tunable When disabled

`Probe_timeout`
The time-out value in seconds that the fault monitor uses to probe an MaxDB database instance.

Data type Integer

Default 90

Range 30–99,999

Tunable Any time

`Restart_if_Parent_Terminated`
Determines whether the fault monitor restarts the MaxDB database instance if the parent kernel process is terminated. The possible values of this extension property are as follows:

- `True` – Specifies that the fault monitor restarts the MaxDB database instance if the parent kernel process is terminated
- `False` – Specifies that the fault monitor does *not* restart the MaxDB database instance if the parent kernel process is terminated

Data type Boolean

Default `False`

Range Not applicable

Tunable Any time

`User_Key`
The user key of the database user that administers the MaxDB database instance. This user key is created when MaxDB is installed and configured as explained in "Installing and Configuring MaxDB" in *Sun Cluster Data Service for MaxDB Guide for Solaris OS* in *Sun Cluster 3.2 Data Service for MaxDB Guide*.

Data type String

Default No default defined

Range Not applicable

Tunable When disabled

Examples

EXAMPLE 1 Creating a `SUNW.sapdb` Resource

```
# clresource create -g sapdbrg -t SUNW.sapdb \\
-p DB_Name=TST -p DB_User=dbadmin -p User_Key=DEFAULT \\
-p resource_dependencies=hsprs sapdbrs
```

This example shows the creation of a `SUNW.sapdb` resource that has the following characteristics:

EXAMPLE 1 Creating a `SUNW.sapdb` Resource *(Continued)*

- The resource is named `sapdbrs`.
- The resource is a member of a resource group that is named `sapdbrg`. The creation of this resource group is not shown in this example.
- The resource is an instance of the `SUNW.sapdb` resource type. The registration of this resource type is not shown in this example.
- The MaxDB database instance that is associated with this resource is named `TST`.
- The UNIX user identity of the OS user that administers the MaxDB database is `dbadmin`.
- The user key of the database user that administers the MaxDB database is `DEFAULT`.
- The MaxDB resource depends on an `HAStoragePlus` resource that is named `hsprs`. The creation of the `hsprs` resource is not shown in this example.

This example does not show the creation of the logical host resource that the MaxDB resource uses.

Attributes

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWscsapdb

See Also

`attributes`(5), `r_properties`(5), SUNW.sap_xserver(5)

`clresourcegroup`(1CL)`clresourcetype`(1CL), `clresource`(1CL),

Sun Cluster Data Service for MaxDB Guide for Solaris OS

Name SUNW.sapenq, sapenq – resource type implementation for the SAP enqueue server component of Sun Cluster HA for SAP Web Application Server

Description The `SUNW.sapenq` resource type represents the SAP enqueue server component in a Sun Cluster configuration. This resource type is part of a set of resource types for the SAP Web Application Server platform. The other components are the SAP replica server (represented by the `SUNW.saprepl` resource type), the SAP message server (represented by the `SUNW.sapscs` resource type), and the SAP web application server component (represented by the `SUNW.sapwebas` resource type).

The SAP enqueue server resource and the SAP message server resource must be in the same failover group (called the SAP central services resource group), because they fail over together. The SAP replica server resource must be in a different failover resource group from the SAP enqueue server resource, because the SAP replica server resource must not fail over with the SAP enqueue server resource.

The resource group affinities must be set to ensure that the SAP central services resource group fails over to the node where the SAP replica resource group has been running and that the SAP replica resource group fails over to another available node.

The resource dependencies must be set to ensure that the SAP replica server resource depends on the SAP enqueue server resource being online.

With the resource group affinities and resource dependencies set as described above, if the SAP enqueue server experiences any hardware or software failure, the SAP central services resource group will fail over to the node where the SAP replica resource group has been running and the SAP replica resource group will fail over to another available node. If the SAP message server experiences any failure, the SAP message server resource will be restarted locally a configurable number of times before a failover is initiated.

Create all these dependencies when you configure the Sun Cluster HA for SAP Web Application Server data service. For more information, see *Sun Cluster Data Service for SAP Web Application Server Guide for Solaris OS*.

Standard properties and extension properties that are defined for the `SUNW.sapenq` resource type are described in the subsections that follow. To set these properties for an instance of the `SUNW.sapenq` resource type, use the `clresourcetype`(1CL) command.

Standard Properties Standard resource properties are overridden for this resource type as follows:

`Retry_Count`
The value of this property must be 0 if the SAP replica server is running. If the standalone SAP enqueue server is running without the SAP replica server, this property can be set to a non-zero value.

Maximum 2

Default 0

Tunable Any time

`Retry_Interval`

Maximum 3600

Default 960

Tunable Any time

`Thorough_Probe_Interval`

Maximum 3600

Default 120

Tunable Any time

For a description of these standard resource properties, see `r_properties`(5).

Extension Properties

The extension properties of this resource type are as follows:

`Child_mon_level`

The child process monitoring level for the process monitor facility (PMF). This property is equivalent to the `-C` option of `pmfadm`.

The default value of `-1` indicates that child process monitoring will not be performed. Positive values indicate the desired level of child process monitoring.

Data type Integer

Default -1

Range No range defined

Tunable Any time

`Enqueue_Instance_Number`

The two-digit instance number for the SAP enqueue server. This is the value of `SAPSYSTEM` in the SAP profile for the SAP enqueue server.

Data type String

Default No default defined

Range Not applicable

Tunable When disabled

`Enqueue_Profile`

The full path to the SAP enqueue server profile.

Data type String

Default No default defined

Range Not applicable

Tunable When disabled

`Enqueue_Server`
The full path to the SAP enqueue server executable.

Data type String

Default No default defined

Range Not applicable

Tunable When disabled

`Enqueue_Server_Monitor`
The full path to the SAP enqueue server monitor executable.

Data type String

Default *directory*`/ensmon`, where *directory* is the full path to the directory where the SAP enqueue server executable is stored, as specified by the extension property `Enqueue_Server`.

Range Not applicable

Tunable When disabled

`Log_Directory`
The directory for the startup and monitor log files.

Data type String

Default The home directory of the administration user, as specified by the extension property `SAP_User`.

Range Not applicable

Tunable When disabled

`Monitor_retry_count`
The maximum number of restarts by the process monitor facility (PMF) that are allowed for the SAP enqueue server fault monitor.

Data type Integer

Default 4

Range No range defined

Tunable Any time

`Monitor_retry_interval`
The interval in minutes between restarts of the SAP enqueue server fault monitor.

Data type Integer

Default 2

Range No range defined

Tunable Any time

`Probe_timeout`
The timeout value in seconds that the SAP enqueue server fault monitor uses to probe an SAP enqueue server instance.

Data type Integer

Default 120

Range Minimum = 2; no maximum defined

Tunable Any time

`SAP_User`
The administration user for the SAP enqueue server.

Data type String, where letters are in lowercase

Default No default defined

Range Not applicable

Tunable When disabled

`Stop_signal`
The signal that is sent to the application to stop the SAP enqueue server application.

Data type Integer

Default 2 (equivalent to `SIGINT`)

Range 1–37

Tunable When disabled

Examples

EXAMPLE 1 Creating Resources for `SUNW.sapenq`, `SUNW.sapscs`, and `SUNW.saprepl`

For this example to work, you must first install the Sun Cluster HA for SAP Web Application Server data service, which includes all the packages to make the SAP Web Application Server components highly available.

The failover SAP central services resource group contains the SAP enqueue server resource, the SAP message server resource, and the logical host resource. The following commands are an example of creating the SAP central services resource group:

```
# clresourcegroup create central-rg
# clreslogicalhostname create -g central-rg -h central-lh \\
```

EXAMPLE 1 Creating Resources for `SUNW.sapenq`, `SUNW.sapscs`, and `SUNW.saprepl` *(Continued)*

```
-N sc_ipmp0@1,sc_ipmp0@2,sc_ipmp0@3,sc_ipmp0@4 central-lh \\
central-lh-rs
```

To bring online the SAP central services resource group, the following command is run:

```
# clresourcegroup -emM central-rg
```

The failover SAP replica resource group contains the SAP replica server resource and a logical host resource. The following commands are an example of creating the SAP replica resource group:

```
# clresourcegroup create repl-rg
# clreslogicalhostname create -g repl-rg -h repl-lh \\
-N sc_ipmp0@1,sc_ipmp0@2,sc_ipmp0@3,sc_ipmp0@4 repl-lh-rs
```

To bring online the SAP replica server resource group, the following command is run:

```
# clresourcegroup -emM repl-rg
```

Setting weak positive resource group affinity between the SAP central services resource group and the SAP replica resource group ensures that, in case of failover, the SAP central services resource group fails over to the node where the SAP replica resource group has been running. The following command is an example of setting this affinity:

```
# clresourcegroup set -p RG_affinities=+repl-rg central-rg
```

The two resource groups must be mastered on different nodes before the strong negative affinity can be set. Therefore, either the SAP central services resource group or the SAP replica resource group must be switched to another node. The following command is an example of switching the SAP central services resource group to another node:

```
# clresourcegroup switch -n node2 central-rg
```

Setting strong negative resource group affinity between the SAP replica resource group and the SAP central services resource group ensures that, in case of failover, after the SAP central services resource group fails over to the node where the SAP replica resource group has been running, the SAP replica resource group will fail over to another available node. The following command is an example of setting this affinity:

```
# clresourcegroup set -p RG_affinities=--central-rg repl-rg
```

To register the resource types, the following commands are run:

```
# clresourcetype register SUNW.sapenq
# clresourcetype register SUNW.sapscs
# clresourcetype register SUNW.saprepl
```

EXAMPLE 1 Creating Resources for `SUNW.sapenq`, `SUNW.sapscs`, and `SUNW.saprepl` *(Continued)*

To create the SAP enqueue server resource in the SAP central services resource group, the following command is run:

```
# clresource create -g central-rg  -t SUNW.sapenq \\
-p Enqueue_Profile=/usr/sap/SC3/SYS/profile/SC3_SCS01_central-lh \\
-p Enqueue_Server=/sapmnt/SC3/exe/enserver \\
-p SAP_User=sc3adm -p Enqueue_Instance_Number=01 enq-rs
```

To create the SAP message server resource in the SAP central services resource group, the following command is run:

```
# clresource create -g central-rg -t SUNW.sapscs \\
-p SAP_SID=SC3 -p SAP_Instance_Number=01 \\
-p SAP_Instance_Name=SCS01 -p Msg_Server_Port=3601 msg-rs
```

To create the SAP replica server resource in the SAP replica resource group, the following command is run:

```
# clresource -g repl-rg -t SUNW.saprepl \\
-p Replica_Profile=/usr/sap/SC3/SYS/profile/SC3_REP01 \\
-p Replica_Server=/sapmnt/SC3/exe/enrepserver \\
-p SAP_User=sc3adm -p Resource_Dependencies=enq-rs repl-rs
```

Attributes

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWscsapenq

See Also

`attributes`(5), `r_properties`(5), SUNW.saprepl(5), SUNW.sapscs(5), SUNW.sapwebas(5)

`clresourcegroup`(1CL)`clresourcetype`(1CL), `clresource`(1CL),

Name SUNW.sap_livecache, sap_livecache – resource type implementation for failover SAP liveCache database

Description The SAP liveCache data service is managed by the Sun Cluster Resource Group Manager (RGM) and is configured as a `LogicalHostname` resource, a SAP liveCache database resource.

The SAP liveCache database depends on the SAP xserver which is managed by data service `SUNW.sap_xserver`. Dependency should be set between the SAP liveCache resource group and the SAP xserver resource group.

You must set the following properties for a SAP liveCache database resource using the `clresource`(1CL) command.

Standard Properties See `r_properties`(5) for a description of the following resource properties.

`Retry_count`	`Default:` 2 `Tunable:` Any time
`Retry_interval`	`Default:` 620 `Tunable:` Any time
`Thorough_probe_interval`	`Default:` 60 `Tunable:` Any time

Extension Properties

`Monitor_retry_count`	Type integer; default is 4. This property controls the restarts of the fault monitor. It indicates the number of times the fault monitor is restarted by the process monitor facility and corresponds to the `-n` option passed to the `pmfadm`(1M) command. The number of restarts is counted in a specified time window (see the property `Monitor_retry_interval`). Note that this property refers to the restarts of the fault monitor itself, not SAP liveCache. SAP liveCache restarts are controlled by the system-defined properties `Thorough_Probe_Interval`, `Retry_Interval`, and `Retry_Count`, as specified in their descriptions. You can modify the value for this property at any time.
`Monitor_retry_interval`	Type integer, default is 2. Indicates the time in minutes over which the failures of the fault monitor are counted and corresponds to the `-t` option passed to the `pmfadm`(1M) command. If the number of times the fault monitor fails exceeds the value of `Monitor_retry_count` within this period, the fault monitor is not restarted by the process monitor facility. You can modify the value for this property at any time.

`Probe_timeout`	Type integer; default is 90. Indicates the time-out value (in seconds) used by the fault monitor to probe a SAP liveCache database instance. You can modify the value for this property at any time.
`Failover_enabled`	Type boolean; defaults to TRUE. Indicates whether to failover or not when `retry_count` is exceeded during `retry_interval`.
`Livecache_Name`	Type string array. This property is the name of the liveCache database instance. Note the name is in uppercase (`LC-NAME`).
`Confdir_list`	Type string array.This property only has one value which is the directory for livecache software and instance directories Default is `/sapdb`.

Examples

EXAMPLE 1 Configuration Example

For this example to work, you must first install the data service.

The following example creates a failover SAP liveCache database resource named `lc-rs` in an existing resource group called `lc-rg`. `lc-rg` must contain a `LogicalHostName` resource.

```
# clresourcetype register SUNW.sap_livecache
# clresource create -g lc-rg -t SUNW.sap_livecache \\
-p LiveCache_Name=LC4 lc-rs
```

In this example, `LC4` is the SAP liveCache database instance name. The rest of the extension properties use the default values.

After the SAP liveCache database resource group and the SAP xserver resource group are created, set the dependency between them.

Attributes

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWsclc

See Also

`pmfadm`(1M), `scha_resource_get`(1HA),`clresourcegroup`(1CL), `clresourcetype`(1CL), `clresource`(1CL), `attributes`(5), `r_properties`(5)

Sun Cluster 3.0 Data Services Installation and Configuration Guide

Name SUNW.saprepl, saprepl – resource type implementation for the SAP replica server component of Sun Cluster HA for SAP Web Application Server

Description The `SUNW.saprepl` resource type represents the SAP replica server component in a Sun Cluster configuration. This resource type is part of a set of resource types for the SAP Web Application Server platform. The other components are the SAP enqueue server (represented by the `SUNW.sapenq` resource type), the SAP message server (represented by the `SUNW.sapscs` resource type), and the SAP web application server component (represented by the `SUNW.sapwebas` resource type).

The SAP enqueue server resource and the SAP message server resource must be in the same failover group (called the SAP central services resource group), because they fail over together. The SAP replica server resource must be in a different failover resource group from the SAP enqueue server resource, because the SAP replica server resource must not fail over with the SAP enqueue server resource.

The resource group affinities must be set to ensure that the SAP central services resource group fails over to the node where the SAP replica resource group has been running and that the SAP replica resource group fails over to another available node.

The resource dependencies must be set to ensure that the SAP replica server resource depends on the SAP enqueue server resource being online.

With the resource group affinities and resource dependencies set as described above, if the SAP enqueue server experiences any hardware or software failure, the SAP central services resource group will fail over to the node where the SAP replica resource group has been running and the SAP replica resource group will fail over to another available node. If the SAP message server experiences any failure, the SAP message server resource will be restarted locally a configurable number of times before a failover is initiated.

Create all these dependencies when you configure the Sun Cluster HA for SAP Web Application Server data service. For more information, see *Sun Cluster Data Service for SAP Web Application Server Guide for Solaris OS*.

Standard properties and extension properties that are defined for the `SUNW.saprepl` resource type are described in the subsections that follow. To set these properties for an instance of the `SUNW.saprepl` resource type, use the `clresourcetype`(1CL) command.

Standard Properties Standard resource properties are overridden for this resource type as follows:

`Retry_Count`

Maximum 2

Default 2

Tunable Any time

`Retry_Interval`

Maximum 3600

Default 620

Tunable Any time

`Thorough_Probe_Interval`

Maximum 3600

Default 120

Tunable Any time

For a description of these standard resource properties, see `r_properties`(5).

Extension Properties The extension properties of this resource type are as follows:

`Child_mon_level`
The child process monitoring level for the process monitor facility (PMF). This property is equivalent to the `-C` option of pmfadm.

The default value of `-1` indicates that child process monitoring will not be performed. Positive values indicate the desired level of child process monitoring.

Data type Integer

Default -1

Range No range defined

Tunable Any time

`Log_Directory`
The directory for the startup and monitor log files that are created by the SAP replica server application.

Data type String

Default The home directory of the administration user, as specified by the extension property `SAP_User`.

Range Not applicable

Tunable When disabled

`Monitor_retry_count`
The maximum number of restarts by the process monitor facility (PMF) that are allowed for the SAP replica server fault monitor.

Data type Integer

Default 4

Range No range defined

Tunable Any time

`Monitor_retry_interval`
The interval in minutes between restarts of the SAP replica server fault monitor.

Data type Integer

Default 2

Range No range defined

Tunable Any time

`Probe_timeout`
Currently unused. The timeout value in seconds that the SAP replica server fault monitor uses to probe an SAP replica server instance. The replica server is started by PMF and monitored by PMF. No additional probing is currently performed by the fault monitor.

Data type Integer

Default 30

Range Minimum = 2; no maximum defined

Tunable Any time

`Replica_Profile`
The full path to the SAP replica server profile.

Data type String

Default No default defined

Range Not applicable

Tunable When disabled

`Replica_Server`
The full path to the SAP replica server executable.

Data type String

Default No default defined

Range Not applicable

Tunable When disabled

`SAP_User`
The administration user for the SAP replica server.

Data type String, where letters are in lowercase

Default No default defined

Range Not applicable

Tunable When disabled

`Stop_signal`
The signal that is sent to the application to stop the SAP replica server application.

Data type Integer

Default 2 (equivalent to `SIGINT`)

Range 1–37

Tunable Any time

Examples

EXAMPLE 1 Creating Resources for `SUNW.sapenq`, `SUNW.sapscs`, and `SUNW.saprepl`

For this example to work, you must first install the Sun Cluster HA for SAP Web Application Server data service, which includes all the packages to make the SAP Web Application Server components highly available.

The failover SAP central services resource group contains the SAP enqueue server resource, the SAP message server resource, and the logical host resource. The following commands are an example of creating the SAP central services resource group:

```
# clresourcegroup create central-rg
# clreslogicalhostname create -g central-rg -l central-lh \\
-N sc_ipmp0@1,sc_ipmp0@2,sc_ipmp0@3,sc_ipmp0@4 central-lh-rs
```

To bring online the SAP central services resource group, the following command is run:

```
# clresourcegroup -emM central-rg
```

The failover SAP replica resource group contains the SAP replica server resource and a logical host resource. The following commands are an example of creating the SAP replica resource group:

```
# clresourcegroup create repl-rg
# clreslogicalhostname create -g repl-rg -l repl-lh \\
-N sc_ipmp0@1,sc_ipmp0@2,sc_ipmp0@3,sc_ipmp0@4 repl-lh-rs
```

To bring online the SAP replica server resource group, the following command is run:

```
# clresourcegroup -emM repl-rg
```

Setting weak positive resource group affinity between the SAP central services resource group and the SAP replica resource group ensures that, in case of failover, the SAP central services resource group fails over to the node where the SAP replica resource group has been running. The following command is an example of setting this affinity:

```
# clresourcegroup set -p RG_affinities=+repl-rg central-rg
```

EXAMPLE 1 Creating Resources for `SUNW.sapenq`, `SUNW.sapscs`, and `SUNW.saprepl` *(Continued)*

The two resource groups must be mastered on different nodes before the strong negative affinity can be set. Therefore, either the SAP central services resource group or the SAP replica resource group must be switched to another node. The following command is an example of switching the SAP central services resource group to another node:

```
# clresourcegroup switch -n Node2 central-rg
```

Setting strong negative resource group affinity between the SAP replica resource group and the SAP central services resource group ensures that, in case of failover, after the SAP central services resource group fails over to the node where the SAP replica resource group has been running, the SAP replica resource group will fail over to another available node. The following command is an example of setting this affinity:

```
# clresourcegroup set -p RG_affinities=--central-rg repl-rg
```

To register the resource types, the following commands are run:

```
# clresourcetype register SUNW.sapenq
# clresourcetype register SUNW.sapscs
# clresourcetype register SUNW.saprepl
```

To create the SAP enqueue server resource in the SAP central services resource group, the following command is run:

```
# clresource create -g central-rg  -t SUNW.sapenq \\
-p Enqueue_Profile=/usr/sap/SC3/SYS/profile/SC3_SCS01_central-lh \\
-p Enqueue_Server=/sapmnt/SC3/exe/enserver \\
-p SAP_User=sc3adm \\
-p Enqueue_Instance_Number=01 sapenq-rs
```

To create the SAP message server resource in the SAP central services resource group, the following command is run:

```
# clresource create -g central-rg -t SUNW.sapscs \\
-p SAP_SID=SC3 -p SAP_Instance_Number=01 \\
-p SAP_Instance_Name=SCS01 \\
-p Msg_Server_Port=3601 msg-rs
```

To create the SAP replica server resource in the SAP replica resource group, the following command is run:

```
#clresource create -g repl-rg -t SUNW.saprepl \\
-p Replica_Profile=/usr/sap/SC3/SYS/profile/SC3_REP01 \\
-p Replica_Server=/sapmnt/SC3/exe/enrepserver \\
-p SAP_User=sc3adm \\
-p Resource_Dependencies=enq-rs repl-rs
```

Attributes See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWscsaprepl

See Also `attributes`(5), `r_properties`(5), SUNW.sapenq(5), SUNW.sapscs(5), SUNW.sapwebas(5)

`clresourcegroup`(1CL)`clresourcetype`(1CL), `clresource`(1CL),

Name SUNW.sapscs, sapscs – resource type implementation for the SAP message server component of Sun Cluster HA for SAP Web Application Server

Description The `SUNW.sapscs` resource type represents the SAP message server component in a Sun Cluster configuration. This resource type is part of a set of resource types for the SAP Web Application Server platform. The other components are the SAP enqueue server (represented by the `SUNW.sapenq` resource type), the SAP replica server (represented by the `SUNW.saprepl` resource type), and the SAP web application server component (represented by the `SUNW.sapwebas` resource type).

The SAP enqueue server resource and the SAP message server resource must be in the same failover group (called the SAP central services resource group), because they fail over together. The SAP replica server resource must be in a different failover resource group from the SAP enqueue server resource, because the SAP replica server resource must not fail over with the SAP enqueue server resource.

The resource group affinities must be set to ensure that the SAP central services resource group fails over to the node where the SAP replica resource group has been running and that the SAP replica resource group fails over to another available node.

The resource dependencies must be set to ensure that the SAP replica server resource depends on the SAP enqueue server resource being online.

With the resource group affinities and resource dependencies set as described above, if the SAP enqueue server experiences any hardware or software failure, the SAP central services resource group will fail over to the node where the SAP replica resource group has been running and the SAP replica resource group will fail over to another available node. If the SAP message server experiences any failure, the SAP message server resource will be restarted locally a configurable number of times before a failover is initiated.

Create all these dependencies when you configure the Sun Cluster HA for SAP Web Application Server data service. For more information, see *Sun Cluster Data Service for SAP Web Application Server Guide for Solaris OS*.

Standard properties and extension properties that are defined for the `SUNW.sapscs` resource type are described in the subsections that follow. To set these properties for an instance of the `SUNW.sapscs` resource type, use the `clresourcetype`(1CL) command.

Standard Properties Standard resource properties are overridden for this resource type as follows:

`Retry_Count`

Maximum 5

Default 2

Tunable Any time

`Retry_Interval`

Maximum 3600

Default 970

Tunable Any time

`Thorough_Probe_Interval`

Maximum 3600

Default 120

Tunable Any time

For a description of these standard resource properties, see `r_properties`(5).

Extension Properties The extension properties of this resource type are as follows:

`Failover_Enabled`
Specifies whether to fail over when `Retry_Count` is exceeded during `Retry_Interval`.

Data type Boolean

Default `TRUE`

Range `TRUE` or `FALSE`

Tunable Any time

`Monitor_Retry_Count`
The maximum number of restarts by the process monitor facility (PMF) that are allowed for the SAP message server fault monitor.

Data type Integer

Default 4

Range No range defined

Tunable Any time

`Monitor_Retry_Interval`
The interval in minutes between restarts of the SAP message server fault monitor.

Data type Integer

Default 2

Range No range defined

Tunable Any time

`Msg_Server_Monitor`
The SAP message server probe executable.

Data type String

Default `/usr/sap/<SAPSID>/SYS/exe/run/msprot`

Range Not applicable

Tunable When disabled

`Msg_Server_Port`
The listen port of the SAP message server.

If no value is specified for this property, the initial default value is 0. In this case a derived default value is calculated to be 3600 + `SAP_Instance_Number`. If the listen port of the SAP message server to be probed is not equivalent to 3600 + `SAP_Instance_Number`, for example, in the case of two SAP message servers, specify a value for this property.

Data type Integer

Default 0

Range 0 – 65535

Tunable When disabled

`Probe_Timeout`
The timeout value in seconds that the SAP message server fault monitor uses to probe an SAP message server instance.

Data type Integer

Default 120

Range Minimum = 2; no maximum defined

Tunable Any time

`SAP_Instance_Name`
The name of the SAP message server instance. This is `INSTANCE_NAME` in the SAP profile.

Data type String

Default None defined

Range Not applicable

Tunable When disabled

`SAP_Instance_Number`
The two-digit SAP system number for the SAP message server instance. This is `SAPSYSTEM` in the SAP profile.

Data type String

Default None defined

Range Not applicable

Tunable When disabled

`SAP_SID`
The SAP system ID. This is `SAPSYSTEMNAME` in the SAP profile.

Data type String

Default None defined

Range Not applicable

Tunable When disabled

`SAP_User`
The administration user for the SAP message server.

Data type String, where letters are in lowercase

Default `<SAPSID>adm`

Range Not applicable

Tunable When disabled

`Scs_Shutdown_Script`
The full path to the shut-down script for the instance.

Data type String

Default `/usr/sap/<SAPSID>/SYS/exe/run/stopsap`

Range Not applicable

Tunable When disabled

`Scs_Startup_Script`
The full path to the start-up script for the instance.

Data type String

Default `/usr/sap/<SAPSID>/SYS/exe/run/startsap`

Range Not applicable

Tunable When disabled

Examples

EXAMPLE 1 Creating Resources for `SUNW.sapenq`, `SUNW.sapscs`, and `SUNW.saprepl`

For this example to work, you must first install the Sun Cluster HA for SAP Web Application Server data service, which includes all the packages to make the SAP Web Application Server components highly available.

EXAMPLE 1 Creating Resources for `SUNW.sapenq`, `SUNW.sapscs`, and `SUNW.saprepl` *(Continued)*

The failover SAP central services resource group contains the SAP enqueue server resource, the SAP message server resource, and the logical host resource. The following commands are an example of creating the SAP central services resource group:

```
# clresourcegroup create central-rg
# clreslogicalhostname create -g central-rg -l central-lh \\
-N sc_ipmp0@1,sc_ipmp0@2,sc_ipmp0@3,sc_ipmp0@4 central-lh-rs
```

To bring online the SAP central services resource group, the following command is run:

```
# clresourcegroup -emM central-rg
```

The failover SAP replica resource group contains the SAP replica server resource and a logical host resource. The following commands are an example of creating the SAP replica resource group:

```
# clresourcegroup create -g repl-rg
# clreslogicalhostname create -g repl-rg -l repl-lh \\
-N sc_ipmp0@1,sc_ipmp0@2,sc_ipmp0@3,sc_ipmp0@4 repl-lh-rs
```

To bring online the SAP replica server resource group, the following command is run:

```
# clresourcegroup -emM repl-rg
```

Setting weak positive resource group affinity between the SAP central services resource group and the SAP replica resource group ensures that, in case of failover, the SAP central services resource group fails over to the node where the SAP replica resource group has been running. The following command is an example of setting this affinity:

```
# clresourcegroup set -p RG_affinities=+repl-rg central-rg
```

The two resource groups must be mastered on different nodes before the strong negative affinity can be set. Therefore, either the SAP central services resource group or the SAP replica resource group must be switched to another node. The following command is an example of switching the SAP central services resource group to another node:

```
# clresourcegroup switch -n Node2 central-rg
```

Setting strong negative resource group affinity between the SAP replica resource group and the SAP central services resource group ensures that, in case of failover, after the SAP central services resource group fails over to the node where the SAP replica resource group has been running, the SAP replica resource group will fail over to another available node. The following command is an example of setting this affinity:

```
# clresourcegroup set -p RG_affinities=--central-rg repl-rg
```

To register the resource types, the following commands are run:

EXAMPLE 1 Creating Resources for `SUNW.sapenq`, `SUNW.sapscs`, and `SUNW.saprepl` *(Continued)*

```
# clresourcetype register SUNW.sapenq
# clresourcetype register SUNW.sapscs
# clresourcetype register SUNW.saprepl
```

To create the SAP enqueue server resource in the SAP central services resource group, the following command is run:

```
# clresource create -g central-rg  -t SUNW.sapenq \\
-p Enqueue_Profile=/usr/sap/SC3/SYS/profile/SC3_SCS01_central-lh \\
-p Enqueue_Server=/sapmnt/SC3/exe/enserver \\
-p SAP_User=sc3adm -p Enqueue_Instance_Number=01 enq-rs
```

To create the SAP message server resource in the SAP central services resource group, the following command is run:

```
# clresource create -g central-rg -t SUNW.sapscs \\
-p SAP_SID=SC3 -p SAP_Instance_Number=01 \\
-p SAP_Instance_Name=SCS01 \\
-p Msg_Server_Port=3601 msg-rs
```

To create the SAP replica server resource in the SAP replica resource group, the following command is run:

```
# clresource create -g repl-rg -t SUNW.saprepl \\
-p Replica_Profile=/usr/sap/SC3/SYS/profile/SC3_REP01 \\
-p Replica_Server=/sapmnt/SC3/exe/enrepserver \\
-p SAP_User=sc3adm \\
-p Resource_Dependencies=enq-rs repl-rs
```

Attributes

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWscsapscs

See Also

`attributes`(5), `r_properties`(5), SUNW.sapenq(5), SUNW.saprepl(5), SUNW.sapwebas(5)

`clresourcegroup`(1CL)`clresourcetype`(1CL), `clresource`(1CL),

Name SUNW.sapwebas, sapwebas – resource type implementation for the SAP web application server component of Sun Cluster HA for SAP Web Application Server

Description The `SUNW.sapwebas` resource type represents the following components in a Sun Cluster Configuration: SAP web application server; J2EE Engine; SAP web Application Server with the J2EE Engine. This resource type is part of a set of resource types for the SAP Web Application Server platform. The other components are the SAP enqueue server (represented by the `SUNW.sapenq` resource type), the SAP replica server (represented by the `SUNW.saprepl` resource type), and the SAP message server (represented by the `SUNW.sapscs` resource type).

The components represented by the SUNW.sapwebas resource may be configured as a failover resource or a scalable resource.

The SAP web application server component resource depends on the database resource being online. The J2EE Engine component depends on the database resource and the SAP message server. You create these dependencies when you configure the Sun Cluster HA for SAP Web Application Server data service and the J2EE Engine data service. For more information, see *Sun Cluster Data Service for SAP Web Application Server Guide for Solaris OS*.

Standard properties and extension properties that are defined for the `SUNW.sapwebas` resource type are described in the subsections that follow. To set these properties for an instance of the `SUNW.sapwebas` resource type, use the `clresourcetype`(1CL) command.

Standard Properties Standard resource properties are overridden for this resource type as follows:

`Retry_Count`

Maximum 5

Default 2

Tunable Any time

`Retry_Interval`

Maximum 4600

Default 4320

Tunable Any time

`Thorough_Probe_Interval`

Maximum 3600

Default 120

Tunable Any time

For a description of these standard resource properties, see `r_properties`(5).

Extension Properties The extension properties of this resource type are as follows:

`Monitor_Retry_Count`
The maximum number of restarts by the process monitor facility (PMF) that are allowed for the SAP web application server component fault monitor.

Data type Integer

Default 4

Range No range defined

Tunable Any time

`Monitor_Retry_Interval`
The interval in minutes between restarts of the SAP web application server component fault monitor.

Data type Integer

Default 2

Range No range defined

Tunable Any time

`Probe_Timeout`
The timeout value in seconds that the SAP web application server component fault monitor uses to probe an SAP web application server component instance.

Data type Integer

Default 120

Range Minimum = 2; no maximum defined

Tunable Any time

`SAP_Instance_Name`
The name of the SAP web application server component instance. This is `INSTANCE_NAME` in the SAP profile.

Data type String

Default None defined

Range Not applicable

Tunable When disabled

`SAP_Instance_Number`
The two-digit SAP system number for the SAP web application server component instance. This is `SAPSYSTEM` in the SAP profile.

Data type String

Default None defined

Range Not applicable

Tunable When disabled

`SAP_SID`
The SAP system ID. This is `SAPSYSTEMNAME` in the SAP profile.

Data type String

Default None defined

Range Not applicable

Tunable When disabled

`SAP_User`
The administration user for the SAP web application server component.

Data type String, where letters are in lowercase

Default `<SAPSID>adm`

Range Not applicable

Tunable When disabled

`SAP_Instance_Type`
The Instance type on the specified Host. The possible values of this extension property are as follows:

- `ABAP` —specifies that SAP Web AS ABAP central instance is installed on the host.
- `J2EE` —specifies that SAP Web AS Java engine is deployed on the host.
- `ABAP_J2EE` —specifies that SAP Web AS ABAP and SAP Web AS Java engine are deployed on the host.

Data type Enum

Default `ABAP`

Range Not applicable

Tunable When disabled

`Log_Directory`
The directory for the startup and monitor log files.

Data type String

Default The home directory of the administration user, as specified by the extension property `SAP_User`.

Range Not applicable

Tunable When disabled

`Webas_Shutdown_Script`
The full path to the shut-down script for the instance.

Data type String

Default `/usr/sap/<SAPSID>/SYS/exe/run/stopsap`

Range Not applicable

Tunable When disabled

`Webas_Startup_Script`
The full path to the start-up script for the instance.

Data type String

Default `/usr/sap/<SAPSID>/SYS/exe/run/startsap`

Range Not applicable

Tunable When disabled

`Webas_Use_Pmf`
Determines if the start-up script process tree is run under Process Monitor Facility (PMF). The possible values of this extension property are as follows:

- `True` - Specifies that the start-up script process tree is run under PMF.
- `False` - Specifies that the start-up script process tree is *not* run under PMF.

Data type Boolean

Default `TRUE`

Range Not applicable

Tunable When disabled

Examples

EXAMPLE 1 Creating a Failover Resource for `SUNW.sapwebas`

For this example to work, you must first install the Sun Cluster HA for SAP Web Application Server data service, which includes all the packages to make the SAP Web Application Server components highly available.

The failover resource group for the SAP web application server component contains the SAP web application server component resource and the logical host resource. The following commands are an example of creating the failover resource group for the SAP web application server component:

```
# clresourcegroup create fo-webas-rg
# clreslogicalhostname create -g fo-webas-rg  -l webas-lh \\
-n sc_ipmp0@1,sc_ipmp0@2,sc_ipmp0@3,sc_ipmp0@4
```

EXAMPLE 1 Creating a Failover Resource for `SUNW.sapwebas` *(Continued)*

To bring online the failover resource group for the SAP web application server component, the following command is run:

```
# clresourcegroup -emM fo-webas-rg
```

To register the resource type, the following command is run:

```
# clresourcetype register SUNW.sapwebas
```

To create a SAP web application server component resource in the failover resource group, the following command is run:

```
# clresource create -g fo-webas-rg -t SUNW.sapwebas \\
-p SAP_SID=SC3 -p SAP_Instance_Number=08 \\
-p SAP_Instance_Name=D08 \\
-p Resource_Dependencies=db-webas-rs,msg-rs webas-rs
```

EXAMPLE 2 Creating a Scalable Resource for `SUNW.sapwebas`

For this example to work, you must first install the Sun Cluster HA for SAP Web Application Server data service, which includes all the packages to make the SAP Web Application Server components highly available.

The scalable resource group for the SAP web application server component contains the SAP web application server component resource and the logical host resource. The following commands are an example of creating the scalable resource group for the SAP web application server component:

```
# clresourcegroup create \\
-p Maximum_primaries=4 \\
-p Desired_primaries=4 \\
sc-webas-rg
# clreslogicalhostname create -g sc-webas-rg -l webas-lh \\
-N sc_ipmp0@1,sc_ipmp0@2,sc_ipmp0@3,sc_ipmp0@4
```

To bring online the scalable resource group for the SAP web application server component, the following command is run:

```
# clresourcegroup -emM sc-webas-rg
```

To register the resource type, the following command is run:

```
# clresourcetype register SUNW.sapwebas
```

To create a SAP web application server component resource in the scalable resource group, the following command is run:

EXAMPLE 2 Creating a Scalable Resource for `SUNW.sapwebas` *(Continued)*

```
# clresource create -g sc-webas-rg -t SUNW.sapwebas \\
-p SAP_SID=SC3 -p SAP_Instance_Number=08 \\
-p SAP_Instance_Name=D08 \\
-p Resource_Dependencies=db-webas-rs,msg-rs webas-rs
```

Attributes

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWscsapwebas

See Also

`attributes`(5), `r_properties`(5), SUNW.sapenq(5), SUNW.sapscs(5), SUNW.saprepl(5)

`clresourcegroup`(1CL),`clresourcetype`(1CL), `clresource`(1CL)

Name SUNW.sap_xserver, sap_xserver – resource type implementation for scalable SAP xserver

Description The SAP xserver data service for Sun Cluster is managed by the Sun Cluster Resource Group Manager (RGM) and is configured as a scalable SAP xserver resource.

You must set the following properties on an SAP xserver resource using the `clresource`(1CL) command.

Standard Properties See `r_properties`(5) for a description of the following resource properties.

`Retry_count`
`Default: 2`

`Tunable:` Any time

`Retry_interval`
`Default: 620`

`Tunable:` Any time

`Thorough_probe_interval`
`Default: 60`

`Tunable:` Any time

Extension Properties `Confdir_list`
Type string. The full path to the directory that contains the software and database instance of the applications that depend on the xserver. These applications can be any combination of the following applications:

- SAP DB
- SAP liveCache

Default is `/sapdb`.

`Independent_Program_Path`
Type string. The full path to the directory that contains the following programs and libraries for SAP xserver:

- Programs that are independent of the database software version
- Libraries for the client runtime environment

A `SUNW.sap_xserver` resource determines the path to the `x_server` command from the value of this property. The `x_server` command resides in the `bin` subdirectory of the directory that this property specifies.

You can modify the value for this property only when the resource is disabled.

`Monitor_retry_count`
Type integer; default is 4. This property controls the restarts of the fault monitor. It indicates the number of times the fault monitor is restarted by the process monitor facility and corresponds to the `-n` option passed to the `pmfadm`(1M) command. The number of

restarts is counted in a specified time window (see the property `Monitor_retry_interval`). Note that this property refers to the restarts of the fault monitor itself, not the SAP xserver. The SAP xserver restarts are controlled by the system-defined properties `Thorough_Probe_Interval` and `Retry_Interval`, as specified in their descriptions. See `clresource`(1CL). You can modify the value for this property at any time.

`Monitor_retry_interval`
Type integer, default is 2. Indicates period of time in minutes during which the PMF counts restarts of the fault monitor and corresponds to the `-t` option passed to the `pmfadm` command. If the number of times the fault monitor fails exceeds the value of `Monitor_retry_count` within this period, the fault monitor is not restarted by the process monitor facility. You can modify the value for this property at any time.

`Probe_timeout`
Type integer; default is 90. Indicates the time-out value (in seconds) used by the fault monitor to probe a SAP xserver instance. You can modify the value for this property at any time.

`Soft_Stop_Pct`
Type integer. This property is the percentage of the `Stop` method timeout that is used to stop SAP xserver by using the SAP utility `x_server stop`. If this timeout is exceeded, the `SIGKILL` signal is used to stop all SAP xserver processes. You can modify the value for this property at any time.

Default is 50.

`Xserver_User`
Type string array. This property is the SAP xserver system administrator user name. You can modify the value for this property only when you have disabled the resource.

Default is `root`.

Examples

EXAMPLE 1 Configuration Example

For this example to work, you must first install the data service.

The following example creates a scalable SAP xserver resource named `xsvr-rs` in a resource group called `xsvr-rg`. The `xsvr-rg` resource group does not contain a `SharedAddress` resource.

```
# clresourcegroup create -p Maximum_primaries=4 \\
-p Desired_primaries=4 xsvr-rg
# clresourcetype register SUNW.sap_xserver
# clresource create -g xsvr-rg -t SUNW.sap_xserver xsrv-rs
```

In this example, SAP xserver will run on 4 cluster nodes. The SAP xserver extension properties use the default values.

Attributes See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWsclc

See Also `pmfadm`(1M), `scha_resource_get`(1HA), `clresourcetype`(1CL),`clresource`(1CL), `clresourcegroup`(1CL) ,`attributes`(5), `r_properties`(5)

Sun Cluster Data Service for MaxDB Guide for Solaris OS, *Sun Cluster Data Service for SAP liveCache Guide for Solaris OS*

Name SUNW.sblgtwy, sblgtwy – resource type implementation for failover Siebel gateway

Description The Siebel data service for Sun Cluster 3.2 is managed by the Sun Cluster Resource Group Manager (RGM) and is configured as a Siebel gateway resource and one or more Siebel server resources.

You must set the following properties for a Siebel gateway resource using the `clresource`(1CL) command.

Standard Properties See `r_properties`(5) for a complete description of the following resource properties.

`Retry_count`
Default: `2`

Tunable: `Any time`

`Retry_interval`
Default: `300`

Tunable: `Any time`

`Thorough_probe_interval`
Default: `60`

Tunable: `Any time`

Extension Properties `Confdir_list`
Type string array. This property is the path name to the Siebel gateway root directory. You can specify the value at resource creation time only.

`Monitor_retry_count`
Type integer. Default is `4`. This property controls the restarts of the fault monitor. It indicates the number of times the fault monitor is restarted by the process monitor facility and corresponds to the `-n` option passed to the `pmfadm`(1M) command. The number of restarts is counted in a specified time window (see the property `Monitor_retry_interval`). Note that this property refers to the restarts of the fault monitor itself, not the Siebel gateway. Siebel gateway restarts are controlled by the system-defined properties `Thorough_Probe_Interval` and `Retry_Interval`, as specified in their descriptions. You can modify the value for this property at any time.

`Monitor_retry_interval`
Type integer. Default is `2`. Indicates the time(in minutes) over which the failures of the fault monitor are counted, and corresponds to the `-t` option passed to the `pmfadm` command. If the number of times the fault monitor fails exceeds the value of `Monitor_retry_count` within this period, the fault monitor is not restarted by the process monitor facility. You can modify the value for this property at any time.

`Probe_timeout`
Type integer Default is `120`. Indicates the time-out value (in seconds) used by the fault monitor to probe a Siebel gateway instance. You can modify the value for this property at

any time.

Examples

EXAMPLE 1 Configuration Example

For this example to work, you must first install the data service.

The following example creates a failover Siebel gateway resource named `sblgtwy-rs` in an existing resource group called `siebel-rg`. `siebel-rg` is assumed to contain a `LogicalHostName` resource.

```
# clresourcetype register SUNW.sblgtwy
# clresource create -g siebel-rg -t SUNW.sblgtwy \
-p Confdir_list=/global/siebel/gtwy sblgtwy-rs
```

In this example, `/global/siebel/gtwy` is the Siebel gateway root directory.

Attributes

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWscsbl

See Also

`pmfadm`(1M), `scha_resource_get`(1HA), `clresourcetype`(1CL), `clresource`(1CL), `clresourcegroup`(1CL), `attributes`(5), `r_properties`(5)

Sun Cluster Data Services Planning and Administration Guide for Solaris OS

Name SUNW.sblsrvr, sblsrvr – resource type implementation for failover Siebel server

Description The Siebel data service for Sun Cluster 3.2 is managed by the Sun Cluster Resource Group Manager (RGM) and is configured as a Siebel gateway resource and one or more Siebel server resources.

You must set the following properties on an Siebel server resource using `clresource`(1CL) command.

Standard Properties See `r_properties`(5) for a description of the following resource properties.

`Retry_count`
Default: `2`

Tunable: `Any time`

`Retry_interval`
Default: `600`

Tunable: `Any time`

`Thorough_probe_interval`
Default: `120`

Tunable: `Any time`

Extension Properties `Confdir_list`
Type string array. This property is the path name to the Siebel server root directory. You can specify the value at resource creation time only.

`Siebel_enterprise`
Type string. This property is set to the name of the Siebel enterprise. You can specify the value at resource creation time only.

`Siebel_server`
Type string. This property is set to the name of the Siebel server. You can specify the value at resource creation time only.

`Monitor_retry_count`
Type integer; default is `4`. This property controls the restarts of the fault monitor. It indicates the number of times the fault monitor is restarted by the process monitor facility and corresponds to the `-n` option passed to the `pmfadm`(1M) command. The number of restarts is counted in a specified time window (see the property `Monitor_retry_interval`). Note that this property refers to the restarts of the fault monitor itself, not the Siebel server. Siebel server restarts are controlled by the system-defined properties `Thorough_Probe_Interval`, `Retry_Interval`, and `Retry_Count`, as specified in their descriptions. You can modify the value for this property at any time.

`Monitor_retry_interval`
Type integer. Default is `2`. Indicates the time in minutes, over which the failures of the fault monitor are counted, and corresponds to the `-t` option passed to the `pmfadm` command. If the number of times the fault monitor fails exceeds the value of `Monitor_retry_count`, the fault monitor is not restarted by the process monitor facility. You can modify the value for this property at any time.

`Probe_timeout`
Type integer. Default is `300`. This property is the time-out value (in seconds) used by the fault monitor to probe a Siebel server instance. You can modify the value for this property at any time.

Examples

EXAMPLE 1 Configuration Example

For this example to work, you must first install the data service.

The following example creates a failover Siebel server resource named `sblsrvr-rs` in an existing resource group called `siebel-rg`. `siebel-rg` is assumed to contain a `LogicalHostName` resource.

```
# clresourcetype register SUNW.sblsrvr
# clresource create -g siebel-rg -t SUNW.sblsrvr \
-p Confdir_list=/global/siebel/srvr \
-p siebel_enterprise=sieb_ent \
-p siebel_server=button-1 sblsrvr-rs
```

In this example, `/global/siebel/srvr` is the Siebel server root directory. The Siebel enterprise is `sieb_ent` and Siebel server name is `button-1`.

Attributes

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWscsbl

See Also

`pmfadm`(1M), `scha_resource_get`(1HA), `clresourcetype`(1CL), `clresource`(1CL), `clresourcegroup`(1CL), `attributes`(5), `r_properties`(5)

Sun Cluster Data Services Planning and Administration Guide for Solaris OS

Name SUNW.scalable_rac_listener, scalable_rac_listener – resource type implementation for the Oracle 9i Real Application Clusters (RAC) listener managed by Sun Cluster

Description The `SUNW.scalable_rac_listener` resource type represents the Oracle 9i RAC listener in a Sun Cluster configuration.

Note – Use the `SUNW.scalable_rac_listener` resource type *only* if you are using Oracle 9i RAC. If you are using Oracle 10g R2, use the SUNW.scalable_rac_server_proxy(5) resource type.

The `SUNW.scalable_rac_listener` resource type is a multiple-master resource type. A single resource of this type can run on multiple nodes concurrently, but does not use network load balancing.

Each `SUNW.scalable_rac_listener` resource represents all Oracle RAC listener instances that serve a database. Each instance of the RAC listener is uniquely identified by the value of the `listener_name` extension property on the node where the instance is running. The `listener_name` extension property is a per-node property. A single resource of this type can take a different value of this property for each node.

For information about how to configure RAC listener resources, see "Configuring Resources for Oracle RAC Database Instances" in *Sun Cluster Data Service for Oracle RAC Guide for Solaris OS* in *Sun Cluster Data Service for Oracle RAC Guide for Solaris OS*.

To register this resource type and create instances of this resource type, use one of the following means:

- Sun Cluster Manager
- The `clsetup`(1CL) utility, specifying the option for configuring Sun Cluster Support for Oracle Real Application Clusters
- The following sequence of Sun Cluster maintenance commands:
  1. To register this resource type, use the `clresourcetype`(1CL) command.
  2. To create instances of this resource type, use the `clresource`(1CL) command.

Standard Properties For a description of all standard resource properties, see the `r_properties`(5) man page.

Standard resource properties are overridden for this resource type as follows:

`Boot_timeout`

Minimum	5
Default	30

`Failover_mode`

Default	None
Tunable	Any time

`Fini_timeout`

Minimum 5

Default 30

`Init_timeout`

Minimum 5

Default 30

`Monitor_start_timeout`

Minimum 5

Default 180

`Monitor_stop_timeout`

Minimum 5

Default 120

`Retry_count`

Minimum -1

Default -1

Tunable Any time

`Retry_interval`

Minimum -1

Maximum 2592000

Default 600

Tunable Any time

`Start_timeout`

Minimum 5

Default 300

`Stop_timeout`

Minimum 5

Default 300

`Thorough_probe_interval`

Minimum 1

Maximum 2592000

Default 30

Tunable Any time

`Update_timeout`

Minimum 5

Default 300

`Validate_timeout`

Minimum 5

Default 120

Extension Properties

The extension properties of the `SUNW.scalable_rac_listener` resource type are as follows.

`debug_level`

This property indicates the level to which debug messages from the Oracle RAC listener component are logged. When the debug level is increased, more debug messages are written to the log files. These messages are logged to the file `/var/opt/SUNWscor/scalable_rac_listener/message_log.`*rs*, where *rs* is the name of the resource that represents the Oracle RAC listener component.

You can specify a different value of the `debug_level` extension property for each node that can master the resource.

Data Type Integer

Range 0–100

Default 1, which logs `syslog` messages

Tunable Any time

`listener_name`

This property specifies the name of the Oracle listener instance that is to be started on the node where the instance is running. This name must match the corresponding entry in the `listener.ora` configuration file.

You can specify a different value of the `listener_name` extension property for each node that can master the resource.

Data Type String

Range Not applicable

Default `LISTENER`

Tunable When disabled

`oracle_home`
This property specifies the full path to the Oracle home directory. The Oracle home directory contains the binary files, log files, and parameter files for the Oracle software.

Data Type	String
Range	Not applicable
Default	No default defined
Tunable	When disabled

`probe_timeout`
This property specifies the timeout value, in seconds, that the fault monitor uses when checking the status of an Oracle RAC listener.

Data Type	Integer
Range	1–99999
Default	300
Tunable	Any time

`user_env`
This property specifies the name of the file that contains the environment variables that are to be set before the listener starts up or shuts down. You must define all environment variables whose values differ from Oracle defaults in this file.

For example, a user's `listener.ora` file might not be located under the `/var/opt/oracle` directory or the *oracle-home*`/network/admin` directory. In this situation, the `TNS_ADMIN` environment variable must be defined.

The definition of each environment variable that is defined must follow the format *variable-name*=*value*. Each definition must start on a new line in the environment file.

You can specify a different value of the `user_env` extension property for each node that can master the resource.

Data Type	String
Range	Not applicable
Default	No default defined
Tunable	Any time

Examples

EXAMPLE 1 Creating a `scalable_rac_listener` Resource

This example shows the commands for performing the following operations to create a `scalable_rac_listener` resource on a two-node cluster:

1. Registering the `SUNW.scalable_rac_server` resource type

EXAMPLE 1 Creating a `scalable_rac_listener` Resource *(Continued)*

2. Adding the `scalable_rac_server-rs` resource to the `rac-db-rg` resource group

A different value of the `listener_name` extension property is set for each node that can master the resource.

The example makes the following assumptions:

- The C shell is used.
- A RAC framework resource group that is named `rac-framework-rg` exists.
- Logical hostname resources that are named `lh1-rs` and `lh2-rs` exist.

```
phys-schost-1# clresourcetype register \
SUNW.scalable_rac_listener
phys-schost-1# clresource create -g rac-db-rg \
-t SUNW.scalable_rac_listener \
-p resource_dependencies_weak=lh1-rs,lh2-rs \
-p oracle_home=/home/oracle/product/9.2.0 \
-p listener_name\{1\}=ORALISTNR1 \
-p listener_name\{2\}=ORALISTNR2 \
scalable_rac_listener-rs
...       Creation of RAC server resource
```

Attributes

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWscor

See Also

`clresource`(1CL), `clresourcetype`(1CL), `clsetup`(1CL), `attributes`(5), `r_properties`(5), `SUNW.rac_cvm`(5), `SUNW.rac_framework`(5), `SUNW.rac_svm`(5), `SUNW.rac_udlm`(5), SUNW.scalable_rac_server(5), SUNW.scalable_rac_server_proxy(5)

Sun Cluster Data Service for Oracle RAC Guide for Solaris OS, *Sun Cluster Data Services Planning and Administration Guide for Solaris OS*

Name SUNW.scalable_rac_server, scalable_rac_server – resource type implementation for the Oracle 9i Real Application Clusters (RAC) server managed by Sun Cluster

Description The `SUNW.scalable_rac_server` resource type represents the Oracle 9i RAC server in a Sun Cluster configuration.

Note – Use the `SUNW.scalable_rac_server` resource type *only* if you are using Oracle 9i RAC. If you are using Oracle 10g R2, use the SUNW.scalable_rac_server_proxy(5) resource type.

The `SUNW.scalable_rac_server` resource type is a multiple-master resource type. A single resource of this type can run on multiple nodes concurrently, but does not use network load balancing.

Each `SUNW.scalable_rac_server` resource represents all Oracle RAC server instances for a database. Each instance of the RAC server is uniquely identified by the value of the `oracle_sid` extension property on the node where the instance is running. The `oracle_sid` extension property is a per-node property. A single resource of this type can take a different value of this property for each node.

Oracle RAC server instances should be started only after the RAC framework is enabled on the cluster node. To ensure that this requirement is met, configure the Oracle RAC server resource and the RAC framework as follows:

- Create a strong positive affinity between the Oracle RAC server resource group and the RAC framework resource group.
- Create a strong dependency between the Oracle RAC server resource and the RAC framework resource.

Create these dependencies and affinities when you configure database resources for the Sun Cluster Support for Oracle RAC data service. For more information, see "Configuring Resources for Oracle RAC Database Instances" in *Sun Cluster Data Service for Oracle RAC Guide for Solaris OS* in *Sun Cluster Data Service for Oracle RAC Guide for Solaris OS*.

To register this resource type and create instances of this resource type, use one of the following means:

- Sun Cluster Manager
- The `clsetup`(1CL) utility, specifying the option for configuring Sun Cluster Support for Oracle Real Application Clusters
- The following sequence of Sun Cluster maintenance commands:
  1. To register this resource type, use the `clresourcetype`(1CL) command.
  2. To create instances of this resource type, use the `clresource`(1CL) command.

Standard Properties For a description of all standard resource properties, see the `r_properties`(5) man page.

Standard resource properties are overridden for this resource type as follows:

`Boot_timeout`

Minimum 5

Default 30

`Failover_mode`

Default `SOFT`

Tunable Any time

`Fini_timeout`

Minimum 5

Default 30

`Init_timeout`

Minimum 5

Default 30

`Monitor_start_timeout`

Minimum 5

Default 120

`Monitor_stop_timeout`

Minimum 5

Default 120

`Retry_count`

Minimum -1

Maximum 9999

Default 2

Tunable Any time

`Retry_interval`

Minimum 0

Maximum 2592000

Default 900

Tunable Any time

`Start_timeout`

Minimum 5

Default 600

`Stop_timeout`

Minimum 5

Default 600

`Thorough_probe_interval`

Minimum 1

Maximum 2592000

Default 30

Tunable Any time

`Update_timeout`

Minimum 5

Default 240

`Validate_timeout`

Minimum 5

Default 120

Extension Properties

The extension properties of the `SUNW.scalable_rac_server` resource type are as follows.

`alert_log_file`

This property is set to the absolute path of the Oracle alert log file. The Oracle software logs alerts in this file. The Oracle RAC server fault monitor scans the alert log file for new alerts at the following times:

- When the RAC server fault monitor is started
- Each time that the RAC server fault monitor queries the health of the server

If an action is defined for a logged alert that the RAC server fault monitor detects, the RAC server fault monitor performs the action in response to the alert.

Preset actions for logged alerts are listed in Appendix B, "Preset Actions for DBMS Errors and Logged Alerts," in *Sun Cluster Data Service for Oracle RAC Guide for Solaris OS* in *Sun Cluster Data Service for Oracle RAC Guide for Solaris OS*. To change the action that the RAC server fault monitor performs, customize the server fault monitor as explained in "Customizing the Oracle 9i RAC Server Fault Monitor" in *Sun Cluster Data Service for Oracle RAC Guide for Solaris OS* in *Sun Cluster Data Service for Oracle RAC Guide for Solaris OS*.

You can specify a different value of the `alert_log_file` extension property for each node that can master the resource.

Data Type String

Range Not applicable

Default `NULL`

Tunable Any time

`connect_cycle`
This property specifies the number of fault monitor probe cycles that are performed before the fault monitor disconnects from the database.

You can specify a different value of the `connect_cycle` extension property for each node that can master the resource.

Data Type Integer

Range 0–99999

Default 5

Tunable Any time

`connect_string`
This property specifies the Oracle database user ID and password that the fault monitor uses to connect to the Oracle database. This property is specified as follows:

userid/*password*

userid
Specifies the Oracle database user ID that the fault monitor uses to connect to the Oracle database.

password
Specifies the password that is set for the Oracle database user *userid*.

The system administrator must define the database user ID and password for the fault monitor during the setup of Oracle RAC. To use Solaris authentication, type a slash (`/`) instead of a user ID and password.

You can specify a different value of the `connect_string` extension property for each node that can master the resource.

Data Type String

Range Not applicable

Default `NULL`

Tunable Any time

`custom_action_file`
This property specifies the absolute path of the file that defines the custom behavior of the Oracle RAC server fault monitor. The format of this file is defined in the custom_action_file(4) man page.

You can specify a different value of the `custom_action_file` extension property for each node that can master the resource.

Data Type	String
Range	Not applicable
Default	Empty string
Tunable	Any time

`debug_level`
This property indicates the level to which debug messages from the Oracle RAC server component are logged. When the debug level is increased, more debug messages are written to the log files. These messages are logged to the file `/var/opt/SUNWscor/scalable_rac_server/message_log.`*rs*, where *rs* is the name of the resource that represents the Oracle RAC server component.

You can specify a different value of the `debug_level` extension property for each node that can master the resource.

Data Type	Integer
Range	0–100
Default	1, which logs `syslog` messages
Tunable	Any time

`oracle_home`
This property specifies the full path to the Oracle home directory. The Oracle home directory contains the binary files, log files, and parameter files for the Oracle software.

Data Type	String
Range	Not applicable
Default	No default defined
Tunable	When disabled

`oracle_sid`
This property specifies the Oracle System Identifier (SID). The Oracle SID uniquely identifies the Oracle RAC database instance on the node where the instance is running.

You *must* specify a different value of the `oracle_sid` extension property for each node that can master the resource.

Data Type String

Range Not applicable

Default `NULL`

Tunable When disabled

`parameter_file`
This property specifies the full path to the Oracle parameter file. This file contains parameters that are to be set when the Oracle database is started. This property is optional. If you do not set this property, the default parameter file that is specified by Oracle is used, namely: *oracle-home*`/dbs/init`*oracle-sid*`.ora`.

oracle-home
Specifies the Oracle home directory

oracle-sid
Specifies the Oracle system identifier of the database instance for which the file is to be used.

You can specify a different value of the `parameter_file` extension property for each node that can master the resource.

Data Type String

Range Not applicable

Default No default defined

Tunable Any time

`probe_timeout`
This property specifies the timeout value, in seconds, that the fault monitor uses when checking the status of an Oracle RAC server.

Data Type Integer

Range 1–99999

Default 300

Tunable Any time

`user_env`
This property specifies the name of the file that contains the environment variables that are to be set before the database starts up or shuts down. You must define all environment variables whose values differ from Oracle defaults in this file.

For example, a user's `listener.ora` file might not be located under the `/var/opt/oracle` directory or the *oracle-home*`/network/admin` directory. In this situation, the `TNS_ADMIN` environment variable must be defined.

The definition of each environment variable that is defined must follow the format *variable-name*=*value*. Each definition must start on a new line in the environment file.

You can specify a different value of the `user_env` extension property for each node that can master the resource.

Data Type String

Range Not applicable

Default No default defined

Tunable Any time

`wait_for_online`

This property specifies whether the `START` method of the Oracle RAC server resource waits for the database to be online before the `START` method exits. The permitted values for this property are as follows:

`True` Specifies that the `START` method of the Oracle RAC server resource waits for the database to be online before the `START` method exits.

`False` Specifies that the `START` method runs the commands to start the database but does not wait for the database to come online before the `START` method exits.

Data Type Boolean

Range Not applicable

Default `True`

Tunable Any time

Examples

EXAMPLE 1 Creating a `scalable_rac_server` Resource

This example shows the commands for performing the following operations to create a `scalable_rac_server` resource on a two-node cluster:

1. Creating the `rac-db-rg` resource group
2. Registering the `SUNW.scalable_rac_server` resource type
3. Adding the `scalable_rac_server-rs` resource to the `rac-db-rg` resource group

A different value of the following extension properties is set for each node that can master the resource:

- `alert_log_file`
- `oracle_sid`

The example makes the following assumptions:

- The C shell is used.

EXAMPLE 1 Creating a `scalable_rac_server` Resource *(Continued)*

- A RAC framework resource group that is named `rac-framework-rg` exists and contains resources of types as shown in the following table:

Resource Type	Resource Name
`SUNW.rac_framework`	`rac_framework-rs`
`SUNW.rac_udlm`	`rac_udlm-rs`
`SUNW.rac_svm`	`rac_svm-rs`

- Creation of a resource of type `SUNW.scalable_rac_listener` that is named `scalable_rac_listener-rs` is outside the scope of this example.

```
phys-schost-1# clresourcegroup create \
-p rg_affinities=++rac-framework-rg \
-p desired_primaries=2 \
-p maximum_primaries=2  \
rac-db-rg
...     Creation of RAC listener resource
phys-schost-1# clresourcetype register SUNW.scalable_rac_server
phys-schost-1# clresource create -g rac-db-rg -t SUNW.scalable_rac_server \
-p resource_dependencies=rac_framework-rs \
-p resource_dependencies_weak=scalable_rac_listener-rs \
-p oracle_home=/home/oracle/product/9.2.0 \
-p connect_string=scooter/t!g3r \
-p oracle_sid\{1\}=V920RAC1 \
-p oracle_sid\{2\}=V920RAC2 \
-p alert_log_file\{1\}=/home/oracle/9.2.0/rdbms/log/alert_V920RAC1.log \
-p alert_log_file\{2\}=/home/oracle/9.2.0/rdbms/log/alert_V920RAC2.log \
scalable_rac_server-rs
```

Attributes

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWscor

See Also

`clresource`(1CL), `clresourcetype`(1CL), `clsetup`(1CL), custom_action_file(4), `attributes`(5), `r_properties`(5), `SUNW.rac_cvm`(5), `SUNW.rac_framework`(5), `SUNW.rac_svm`(5), `SUNW.rac_udlm`(5), SUNW.scalable_rac_listener(5), SUNW.scalable_rac_server_proxy(5)

Sun Cluster Data Service for Oracle RAC Guide for Solaris OS, *Sun Cluster Data Services Planning and Administration Guide for Solaris OS*

Name SUNW.scalable_rac_server_proxy, scalable_rac_server_proxy – resource type implementation for the Oracle Real Application Clusters (RAC) server proxy managed by Sun Cluster

Description The `SUNW.scalable_rac_server_proxy` resource type represents a proxy for the Oracle 10g R2 RAC server in a Sun Cluster configuration.

Note – Use the `SUNW.scalable_rac_server_proxy` resource type *only* if you are using Oracle 10g R2 RAC. If you are using Oracle 9i, use the SUNW.scalable_rac_server(5) resource type.

In Oracle 10g, Oracle Cluster Ready Services (CRS) manage the startup and shutdown of RAC server instances. To be managed by the CRS, these instances must be registered with the CRS. The Oracle CRS software also provides automatic fault monitoring and failure recovery for RAC server instances. These instances are represented as resources to Oracle CRS.

A resource of type `SUNW.scalable_rac_server_proxy` is a *proxy* resource: The proxy resource acts as a substitute for a resource that is managed by Oracle CRS. The proxy resource enables Sun Cluster utilities to manage Oracle RAC server instances *through* Oracle CRS. In this way, the `SUNW.scalable_rac_server_proxy` resource type enables the clustering frameworks that are provided by Sun Cluster and Oracle Cluster Ready Services (CRS) to interoperate.

The `SUNW.scalable_rac_server_proxy` resource type enables you to use Sun Cluster utilities as an alternative to Oracle utilities to start and stop Oracle RAC database instances.

Each resource of type `SUNW.scalable_rac_server_proxy` has a monitor that obtains the following status information from the CRS resource for which the `SUNW.scalable_rac_server_proxy` resource is acting as a proxy.

- The online or offline status of Oracle CRS
- The status of an Oracle RAC database instance:
  - Online and enabled
  - Online but disabled
  - Offline and disabled
- The success or failure of an attempt to start or stop a database instance
- The ability of Oracle CRS to manage the Oracle RAC database instance

  The monitor probes the Oracle CRS determine if the CRS are managing the RAC database instance. If the CRS does not indicate that the CRS are managing the RAC database instance, the monitor warns that the instance is invalid. However, the Oracle CRS might not be managing the RAC database instance because the instance is unregistered with the CRS. In this situation, the RAC database instance might be valid, despite the warning.

The timeout period that the monitor uses for obtaining status information is determined by the `proxy_probe_timeout` extension property. If the timeout period is too short, timeouts

might cause the monitor to report the status of a valid RAC database instance as invalid. In this situation, consider increasing the value of the he `proxy_probe_timeout` extension property.

This monitor only enables the status of Oracle RAC database instances to be monitored by Sun Cluster utilities. This monitor does *not* provide fault monitoring and automatic fault recovery for Oracle RAC database instances. The Oracle CRS software provides this functionality.

Oracle RAC server instances should be started only after the RAC framework and any storage resources are enabled on the cluster node. To ensure that this requirement is met, configure the Oracle RAC server proxy resource as follows:

- Create a strong positive affinity between the Oracle RAC server proxy resource group and the following resource groups:
  - The RAC framework resource group
  - Any resource group that contains storage resources for Oracle files
- Create a strong dependency between the Oracle RAC server proxy resource and the RAC framework resource.
- Create an offline-restart dependency between the Oracle RAC server proxy resource and the following resources:
  - The CRS framework resource
  - Any storage resources for Oracle files that you are using

Create these dependencies and affinities when you configure database resources for the Sun Cluster Support for Oracle RAC data service. For more information, see "Configuring Resources for Oracle RAC Database Instances" in *Sun Cluster Data Service for Oracle RAC Guide for Solaris OS* in *Sun Cluster Data Service for Oracle RAC Guide for Solaris OS*.

To register this resource type and create instances of this resource type, use one of the following means:

- Sun Cluster Manager
- The `clsetup`(1CL) utility, specifying the option for configuring Sun Cluster Support for Oracle Real Application Clusters
- The following sequence of Sun Cluster maintenance commands:
  1. To register this resource type, use the `clresourcetype`(1CL) command.
  2. To create instances of this resource type, use the `clresource`(1CL) command.

Note – A Solaris project might be specified for a `SUNW.scalable_rac_server_proxy` resource or the resource group that contains a `SUNW.scalable_rac_server_proxy` resource. In this situation, the project affects *only* the processes for the `SUNW.scalable_rac_server_proxy` resource. The project does *not* affect the processes for any resources that Oracle CRS control, including processes for RAC database instances.

Standard Properties

For a description of all standard resource properties, see the `r_properties`(5) man page.

Standard resource properties are overridden for this resource type as follows:

`Boot_timeout`

Minimum 5

Default 30

`Failover_mode`

Default None

Tunable Any time

`Fini_timeout`

Minimum 5

Default 30

`Init_timeout`

Minimum 5

Default 30

`Load_balancing_policy`

Default `LB_weighted`

Tunable At creation

`Load_balancing_weights`

Default Empty string

Tunable Any time

`Network_resources_used`

Tunable When disabled

Default Empty string

`Port_list`

Default None

Tunable At creation

`Retry_Count`

Maximum 10

Default 2

Tunable Any time

`Retry_Interval`

Maximum 3600

Default 300

Tunable Any time

`Start_timeout`

Minimum 5

Default 600

`Stop_timeout`

Minimum 5

Default 600

`Thorough_probe_interval`

Minimum 1

Maximum 2592000

Default 20

Tunable Any time

`Update_timeout`

Minimum 5

Default 240

`Validate_timeout`

Minimum 5

Default 120

Extension Properties

The extension properties of the `SUNW.scalable_rac_server_proxy` resource type are as follows.

`client_retries`

This property specifies the maximum number of attempts by the resource's remote procedure call (RPC) client to connect to the proxy daemon.

Data Type Integer

Range 1–25

Default 3

Tunable When disabled

`client_retry_interval`

This property specifies the interval, in seconds, between attempts by the resource's remote procedure call (RPC) client to connect to the proxy daemon.

Data Type Integer

Range 1–25

Default 5

Tunable When disabled

`crs_home`

This property specifies the directory in which the Oracle CRS software is located.

Data Type String

Range Not applicable

Default No default defined

Tunable When disabled

`db_name`

This property specifies the name that uniquely identifies the specific Oracle RAC database that is associated with this resource. This identifier distinguishes the database from other databases that might run simultaneously on your system. The name of the Oracle RAC database is specified during the installation of Oracle RAC.

Data Type String

Range Not applicable

Tunable When disabled

`debug_level`

This property indicates the level to which debug messages from the monitor for the Oracle RAC proxy server are logged. When the debug level is increased, more debug messages are written to the log files.

The messages are logged to files in the directory `/var/opt/SUNWscor/oracle_server/proxy`*rs*. Messages for server-side components and client-side components of the proxy server resource are written to separate files:

- Messages for server-side components are written to the file `message_log.`*rs*.
- Messages for client-side components are written to the file `message_log.client.`*rs*.

In these file names and directory names, *rs* is the name of the resource that represents the Oracle RAC server component.

You can specify a different value of the `debug_level` extension property for each node that can master the resource.

Data Type Integer

Range 0–100

Default 1, which logs `syslog` messages

Tunable Any time

`monitor_probe_interval`
This property specifies the interval, in seconds, between probes of the CRS resource for which this resource is acting as a proxy.

Data Type Integer

Range 10–600

Default 120

Tunable Any time

`oracle_home`
This property specifies the full path to the Oracle home directory. The Oracle home directory contains the binary files, log files, and parameter files for the Oracle software.

Data Type String

Range Not applicable

Default No default defined

Tunable When disabled

`oracle_sid`
This property specifies the Oracle System Identifier (SID). The Oracle SID uniquely identifies the Oracle RAC database instance on the node where the instance is running.

You *must* specify a different value of the `oracle_sid` extension property for each node that can master the resource. The value for each node must correctly identify the instance that is running on the node.

Data Type String

Range Not applicable

Default `NULL`

Tunable When disabled

`proxy_probe_timeout`
This property specifies the timeout value, in seconds, that the proxy monitor uses when checking the status of the CRS resource for which this resource is acting as a proxy.

Data Type	Integer
Range	5–100
Default	25
Tunable	Any time

`startup_wait_count`
This property specifies the maximum number of attempts by this resource to confirm that the Oracle CRS software is started completely. The interval between attempts is twice the value of the `proxy_probe_timeout` extension property.

The resource requires confirmation that Oracle CRS software is started before attempting to start an Oracle RAC database instance. If the maximum number of attempts is exceeded, the resource does not attempt to start the database instance.

Data Type	Integer
Range	10–600
Default	20
Tunable	When disabled

`user_env`
This property specifies the name of the file that contains the environment variables that are to be set before the database starts up or shuts down. You must define all environment variables whose values differ from Oracle defaults in this file.

For example, a user's `listener.ora` file might not be located under the `/var/opt/oracle` directory or the *oracle-home*`/network/admin` directory. In this situation, the `TNS_ADMIN` environment variable must be defined.

The definition of each environment variable that is defined must follow the format *variable-name*=*value*. Each definition must start on a new line in the environment file.

You can specify a different value of the `user_env` extension property for each node that can master the resource.

Data Type	String
Range	Not applicable
Default	No default defined
Tunable	Any time

Examples

EXAMPLE 1 Creating a `scalable_rac_server_proxy` Resource

This example shows the commands for performing the following operations to create a `scalable_rac_server_proxy` resource on a two-node cluster:

EXAMPLE 1 Creating a `scalable_rac_server_proxy` Resource *(Continued)*

1. Creating the `rac-proxy-db-rg` resource group
2. Registering the `SUNW.scalable_rac_server_proxy` resource type
3. Adding the `scalable_rac_server_proxy-rs` resource to the `rac-proxy-db-rg` resource group

A different value of the `oracle_sid` extension property is set for each node that can master the resource.

The example makes the following assumptions:

- The C shell is used.
- A RAC database that is named `V1020RAC` is registered with the Oracle CRS.
- A resource group that is named `scal-dg-rg` exists and contains a resource of type `SUNW.ScalDeviceGroup` that is named `scal-dg-rs`.
- A RAC framework resource group that is named `rac-framework-rg` exists and contains resources that are shown in the following table:

Resource Type	Resource Name
`SUNW.crs_framework`	`crs_framework-rs`
`SUNW.rac_framework`	`rac_framework-rs`
`SUNW.rac_udlm`	`rac_udlm-rs`
`SUNW.rac_svm`	`rac_svm-rs`

```
phys-schost-1# clresourcegroup create \
-p rg_affinities=++rac-framework-rg,++scal-dg-rg \
-p desired_primaries=2 \
-p maximum_primaries=2  \
rac-proxy-db-rg
phys-schost-1# clresourcetype register \
SUNW.scalable_rac_server_proxy
phys-schost-1# clresource create -g rac-proxy-db-rg \
-t SUNW.scalable_rac_server_proxy \
-p resource_dependencies=rac_framework-rs \
-p resource_dependencies_offline_restart=crs-framework-rs,scal-dg-rs \
-p oracle_home=/home/oracle/product/10.2.0/oracle_install \
-p crs_home=/home/oracle/product/10.2.0/crs_install \
-p db_name=V1020RAC \
-p oracle_sid\{1\}=V1020RAC1 \
-p oracle_sid\{2\}=V1020RAC2 \
scalable_rac_server_proxy-rs
```

Attributes

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWscor

See Also

`clresource`(1CL), `clresourcetype`(1CL), `clsetup`(1CL), `attributes`(5), `r_properties`(5), `SUNW.rac_cvm`(5), `SUNW.rac_framework`(5), `SUNW.rac_svm`(5), `SUNW.rac_udlm`(5), SUNW.scalable_rac_server(5)

Sun Cluster Data Service for Oracle RAC Guide for Solaris OS, *Sun Cluster Data Services Planning and Administration Guide for Solaris OS*

Name SUNW.sybase, sybase – resource type implementation for Sun Cluster HA for Sybase Adaptive Server Enterprise (ASE)

Description The `SUNW.sybase` resource type represents the Sybase ASE application in a Sun Cluster configuration.

Standard properties and extension properties that are defined for the `SUNW.sybase` resource type are described in the subsections that follow. To set these properties for an instance of the `SUNW.sybase` resource type, use the `clresource`(1CL) command or a resource-configuration GUI.

Standard Properties Set the standard resource property `Failover` for all failover resource types.

Standard resource properties are overridden for this resource type as follows:

`Failover_mode`
Default: `SOFT`

Tunable: Any time

`Retry_count`
Default: 2

Tunable: Any time

`Retry_interval`
Default: 600

Tunable: Any time

`Thorough_probe_interval`
Default: 30

Tunable: Any time

For a description of these standard resource properties, see `r_properties`(5).

Extension Properties `Adaptive_Server_Log_File`
Type string. Set this property as the absolute path of the Sybase ASE data-server log to which Sybase ASE logs errors. As part of the probe execution, the Sybase ASE data-server fault monitor scans this log file for errors. The fault monitor matches the error numbers for possible actions with patterns that the Sun Cluster HA for Sybase ASE action file, `/opt/SUNWsyb/etc/sybase_actions`, specifies. You can modify this property at any time. No default value exists for this field. You must set this property.

`Adaptive_Server_Name`
Type string. This property specifies the adaptive-server name, which enables the Sun Cluster HA for Sybase ASE data service to locate and execute the `RUN` server script. This script is located in the `$SYBASE/ASE_`*major-version*`/install` directory, where *major-version* is the major version of Sybase ASE that you are using. For example, if you are

using Sybase ASE version 12.5.1, *major-version* is `12-5`. You can modify this property only when you have disabled the resource. No default value exists for this field. You must set this property.

`Backup_Server_Name`
Type string. This property specifies the backup-server name, which enables the Sun Cluster HA for Sybase ASE data service to locate and execute the `RUN` server script. This script is located in the `$SYBASE/ASE_`*major-version*`/install` directory, where *major-version* is the major version of Sybase ASE that you are using. For example, if you are using Sybase ASE version 12.5.1, *major-version* is `12-5`. You can modify this property only when you have disabled the resource. Setting this property is optional, but if you do not set the property, the Sun Cluster HA for Sybase ASE data service will not manage the server.

`Connect_cycle`
Type integer. Default is 5. The Sybase ASE data-server fault monitor uses the user ID and password that the `Connect_string` property specifies to periodically connect to the database. After executing the number of probes that this property specifies, the monitor disconnects and then reconnects. You can modify the value for this property at any time.

`Connect_string`
Type string. Set this property to the database user's user ID and password in fault-monitor transactions. Specify this property as follows:

userid/*password*

When you set up the Sun Cluster HA for Sybase ASE data service, define the database user ID and password before you enable the server resource and the server resource's fault monitor. Do *not* use the `sa` account for the database user. You can modify this property at any time. No default value exists for this field. You must set this property, even if you do not set the `Monitor_Server_Name` property.

`Debug_level`
Type integer. Default is 1. This property indicates the debug level for writing to the Sun Cluster HA for Sybase ASE log. You can modify the value for this property at any time.

`Environment_File`
Type string. This property specifies the absolute file path of the environment file (typically `SYBASE.sh`) that is provided with the Sybase ASE distribution. Before executing any method or program, the Sun Cluster HA for Sybase ASE data service reads this file and sets the environment accordingly. You can modify this property only when you have disabled the resource. No default value exists for this field. You must set this property.

`Monitor_Server_Name`
Type string. This property specifies the monitor-server name, which enables the Sun Cluster HA for Sybase ASE data service to locate and execute the `RUN` server script. This script is located in the `$SYBASE/ASE_`*major-version*`/install` directory, where *major-version* is the major version of Sybase ASE that you are using. For example, if you are using Sybase ASE version 12.5.1, *major-version* is `12-5`. You can modify this property only

when you have disabled the resource. Setting this property is optional, but if you do not set the property, the Sun Cluster HA for Sybase ASE data service will not manage the server.

`Probe_timeout`
Type integer. Default is 60 seconds. This property is the timeout value that the fault monitor uses to probe a Sybase ASE server instance. You can modify the value for this property at any time.

`Stop_File`
Type string. This property indicates the absolute path to the script that the `STOP` method executes to stop the Sybase ASE servers. This file stores the password of the Sybase ASE system administrator (`sa`). Protect the path so that only the user and group that are associated with the Sybase ASE installation can access the file. The Sun Cluster HA for Sybase ASE package includes the `sybase_stop_servers` template. You must replace the existing password. You can modify this property at any time. No default value exists for this field. You must set this property.

`Text_Server_Name`
Type string. This property specifies the text-server name, which enables the Sun Cluster HA for Sybase ASE data service to locate and execute the `RUN` server script. This script is located in the `$SYBASE/ASE_`*major-version*`/install` directory, where *major-version* is the major version of Sybase ASE that you are using. For example, if you are using Sybase ASE version 12.5.1, *major-version* is `12-5`. You can modify this property only when you have disabled the resource. Setting this property is optional, but if you do not set the property, the Sun Cluster HA for Sybase ASE data service will not manage the server.

`Wait_for_online`
Type Boolean. Default is `TRUE`. This property specifies whether the `START` method waits for the database to become active before exiting. If you set this property to `TRUE`, the `START` method starts the database and waits for the database to become active before exiting. You can modify the value for this property at any time.

Attributes

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWscsyb

See Also

`pmfadm`(1M), `scha_resource_get`(1HA), `clresource`(1CL), `clresourcegroup`(1CL), `attributes`(5), `r_properties`(5)

Sun Cluster Data Services Planning and Administration Guide for Solaris OS, *Sun Cluster Data Service for Sybase ASE Guide for Solaris OS*

Name SUNW.wls, wls – resource type implementation for failover BEA WebLogic Server

Description The Resource Group Manager (RGM) manages Sun Cluster HA for BEA WebLogic Server for Sun Cluster. Use the `clresource`(1CL) command or a resource configuration GUI to set the following properties on BEA WebLogic Server resources.

Standard Properties See `r_properties`(5) for a complete description of the following resource properties.

`Failover_mode`

Default `SOFT`

Tunable Any time

`Probe_timeout`

Default 180

Tunable Any time

`Retry_count`

Default 2

Tunable Any time

`Retry_interval`

Default 480

Tunable Any time

`Thorough_probe_interval`

Default 120

Tunable Any time

Extension Properties `Confdir_list`

Type string array. No default value exists. Use this property to set the complete path to the BEA WebLogic Server home directory, `$WL_HOME`.

You can modify this property only when you create the resource.

`DB_Probe_Script`

Type string. Default is null. This extension property can be used to set the complete path to a database probe script. The `HA-WLS` probe method probes only the `WLS` instances. The database probe script can be provided by using this extension property if the administrators want the `HA_WLS` to probe the DB also. This probe script must return 0 for success. The BEA WebLogic Servers are started only if this database probe returns success. If an `HA-WLS` probe detects a failure in a WLS instance and if this extension property is set, the `HA-WLS` probe takes action only if the database probe succeeds.

You can modify this property at any time.

`Monitor_Uri_List`
Type string. Default is null. This property indicates the URI or list of URIs, separated by a comma, that can be used by the fault monitor to test the functionality of the BEA WebLogic Server by running an `HTTP GET` command on the URI. The `Server_Url` extension property is for simple probes on the hostname and port. This extension property can be used to probe the WLS functionality by retrieving a Java servlet or making the WLS run an application and return an HTTP server code. If the HTTP server return code is 500 (Internal Server Error) or if the connection fails, the probe takes action. See the probe method for more details.

Make sure that the hostnames used in the `server_url` and `monitor_uri_list` are resolvable. If you use Fully Qualified Domain Names (FQDNs), then DNS must be enabled and `/etc/nsswitch.conf` must have the correct entries to resolve the hostnames by using DNS.

You can modify this property at any time.

`Server_name`
Type string. Default is null. A single start script can be used to start all the Managed Servers by passing the Managed Server name as an argument to the START script. If Agent START methods have to pass the server name as an argument to the START script, configure the Managed Server name in this extension property.

You can modify this property at any time.

`Server_Url`
Type string. No default value exists. This property indicates the URL of the BEA WebLogic Server. The URL includes the protocol that is used to connect to the server, that this resource is configured to start. The probe method uses this URL to check the health of the server by running an `HTTP GET` command on the URL. The protocol in the URL must be set to http. The complete URL should be in the following form.

```
http://host:port
```

Make sure that the hostnames used in the `server_url` and `monitor_uri_list` are resolvable. If you use Fully Qualified Domain Names (FQDNs), then DNS must be enabled and `/etc/nsswitch.conf` must have the correct entries to resolve the hostnames by using DNS.

You can modify this property only when you create the resource.

`Smooth_shutdown`
Type Boolean. Default is `False`. This extension property can be used to enable smooth shutdown by using the `WebLogic.Admin` class. This extension property must be set to TRUE if a smooth shutdown is desired before trying to kill the WLS process. If this extension property is TRUE, the `WLS_USER` and `WLS_PW` must be set in the `start_script` and not in `boot.properties`.

Set this extension property to TRUE if all of the following apply.

- Setting the `username` and `password` in the `start_script` is not a concern.
- A smooth shutdown is desirable instead of the default, killing the process.
- You are not concerned if the agent STOP method reads the user name and password from the START script and passes it to the `java weblogic.Admin` command.

You can modify this property at any time.

`Start_Script`
Type string. No default value exists. Use this property to set the complete path to the script that should be used to start the BEA WebLogic Server instance (either Administration or Managed). This script is typically present under the domain name directory along with the `config.xml` file. A separate script or a single script can be configured for starting each BEA WebLogic Server.

You can modify this property only when you create the resource.

Examples

EXAMPLE 1 Creating a Simple BEA WebLogic Server Resource

This example assumes that the START script, `startWebLogic.sh`, can start the BEA WebLogic Server without any arguments to the script. The `username` and `password` needed to start the BEA WebLogic Server can be configured within this START script or in the `boot.properties` file.

```
clresource create -g bea-rg -t SUNW.wls \
-p Confdir_list=/global/bea/beahome/weblogic700 \
-p Server_url=http://logical-host-1:7001 \
-p Start_script=/global/bea/beahome/user_projects/ha-wls/startWebLogic.sh bea-rs
```

EXAMPLE 2 Creating a Managed Server Resource Whose Start Script Takes a Managed Server Name as an Input

For this example to work, the Admin_URL must be set within the `Start_script` `startManagedWebLogic.sh`.

```
clresource create -g bea-rg -t SUNW.wls \
-p Confdir_list=/global/bea/beahome/weblogic700 \
-p Server_url=http://logical-host-1:7004 \\
-p Start_script=/global/bea/beahome/user_projects/ha-wls/startManagedWebLogic.sh \
-p Server_name=test1 bea-rs1
```

EXAMPLE 3 Creating a BEA WebLogic Server Managed Server Resource Which Should Be Shut Down Smoothly

This example creates a resource that has the extension property `Smooth_shutdown` set to TRUE. Setting this extension property to TRUE specifies that BEA WebLogic Server will shut down the resource smoothly. If this extension property is not set to TRUE, the `STOP` method sends `sigkill` to the BEA WebLogic Server. For the resource created in this example, the BEA

EXAMPLE 3 Creating a BEA WebLogic Server Managed Server Resource Which Should Be Shut Down Smoothly *(Continued)*

WebLogic Server first tries a smooth shutdown by using the `Weblogic.Admin` class. If this attempt is not successful, BEA WebLogic Server uses `sigkill`. The `WLS_PW` and `WLS_USER` must be set in the `START` script `startManagedWeblogic.sh`. If these two parameters are not set in the `START` script, the resource creation fails.

```
clresource create -g bea-rg -t SUNW.wls \
-p Confdir_list=/global/bea/beahome/weblogic700 \
-p Server_url=http://logical-host-1:7008 \
-p Start_script=/global/bea/beahome/user_projects/ha-wls/startManagedWebLogic.sh \
-p Server_name=text2 -p smooth_shutdown=true bea-rs2
```

EXAMPLE 4 Creating a BEA WebLogic Server Resource that Should Probe the Database Before Taking Any Action on the BEA WebLogic Server

This example creates a BEA WebLogic Server resource that should probe the database before taking any action on the BEA WebLogic Server. If the `db_probe` script is set, the BEA WebLogic Server will not be started if the script returns a failure. When the BEA WebLogic Server probe fails, action is taken only if the `db_probe_script` returns 0. This user-supplied database probe script must return 0 for success and non zero for failure.

```
clresource create -g bea-rg -t SUNW.wls \
-p Confdir_list=/global/bea/beahome/weblogic700 \
-p Server_url=http://logical-host-1:710 \
-p Start_script=/global/bea/beahome/user_projects/ha-wls/startManagedWebLogic.sh \
-p Server_name=test3 \
-p db_probe_script=/global/phys-pale-1/bea-db_probe_script bea-rs3
```

EXAMPLE 5 Creating a BEA WebLogic Server Resource that Should Also Monitor Some URIs Along With the Server_url Monitoring

This example creates a BEA WebLogic Server resource that monitors URIs by using the `Monitor_uri_list` extension property. Action is taken only if the URI returns an http error 500 or if the connection to the BEA WebLogic Server fails.

```
clresource create -g bea-rg -t SUNW.wls \
-p Confdir_list=/global/bea/beahome/weblogic700 \
-p Server_url=http://logical-host-1:7012 \
-p Start_script=/global/bea/beahome/user_projects/ha-wls/startManagedWebLogic.sh \
-p Server_name=test5 \
-p db_probe_script=/global/bea/db_probe_script \
-p monitor_uri_list=http://logical-host-1:7001/sctest bea-rs5
```

Attributes See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWwls

See Also `attributes`(5)

`clresource`(1CL)

`r_properties`(5)

`scha_resource_get`(3HA)

Sun Cluster Data Service for WebLogic Server Guide for Solaris OS

Index

F

H

I

J

K

L

M

N

O

P

R

S

T

U

W

X