Oracle® Clusterware

Administration and Deployment Guide

12*c* Release 1 (12.1)

E48819-09

July 2015

Oracle Clusterware Administration and Deployment Guide, 12*c* Release 1 (12.1)

E48819-09

Primary Author: Richard Strohm

Contributor: The Oracle Database 12*c* documentation is dedicated to Mark Townsend, who was an inspiration to all who worked on this release.

Contributors: Ahmed Abbas, Troy Anthony, Ram Avudaiappan, Mark Bauer, Eric Belden, Suman Bezawada, Gajanan Bhat, Burt Clouse, Jonathan Creighton, Mark Fuller, Apostolos Giannakidis, Angad Gokakkar, John Grout, Andrey Gusev, Winston Huang, Sameer Joshi, Sana Karam, Roland Knapp, Erich Kreisler, Raj K. Kammend, Karen Li, Barb Lundhild, Bill Manry, Saar Maoz, John McHugh, Markus Michalewicz, Anil Nair, Siva Nandan, Philip Newlan, Srinivas Poovala, Sampath Ravindhran, Kevin Reardon, Dipak Saggi, K.P. Singh, Duane Smith, Janet Stern, Su Tang, Douglas Williams, Soo Huey Wong

Contents

3 Policy-Based Cluster and Capacity Management

4 Oracle Flex Clusters

5 Rapid Home Provisioning

6 Managing Oracle Cluster Registry and Voting Files

7 Adding and Deleting Cluster Nodes

8 Cloning Oracle Clusterware

9 Making Applications Highly Available Using Oracle Clusterware

A Cluster Verification Utility Reference

B Oracle Clusterware Resource Reference

C OLSNODES Command Reference

D Oracle Interface Configuration Tool (OIFCFG) Command Reference

E Oracle Clusterware Control (CRSCTL) Utility Reference

G Oracle Clusterware Agent Framework C Application Program Interfaces

H Oracle Clusterware C Application Program Interfaces

I Oracle Cluster Registry Utility Reference

J Troubleshooting Oracle Clusterware

List of Tables

Preface

The *Oracle Clusterware Administration and Deployment Guide* describes the Oracle Clusterware architecture and provides an overview of this product. This book also describes administrative and deployment topics for Oracle Clusterware.

Information in this manual applies to Oracle Clusterware as it runs on all platforms unless otherwise noted. In addition, the content of this manual supplements administrative and deployment topics for Oracle single-instance databases that appear in other Oracle documentation. Where necessary, this manual refers to platform-specific documentation. This Preface contains these topics:

- Audience
- Documentation Accessibility
- Related Documents
- Conventions

Audience

The *Oracle Clusterware Administration and Deployment Guide* is intended for database administrators, network administrators, and system administrators who perform the following tasks:

- Install and configure Oracle Real Application Clusters (Oracle RAC) databases
- Administer and manage Oracle RAC databases
- Manage and troubleshoot clusters and networks that use Oracle RAC

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc`.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

Related Documents

For more information, see the Oracle resources listed in this section.

- Platform-specific Oracle Clusterware and Oracle RAC installation guides

  Each platform-specific Oracle Database 11*g* installation media contains a copy of an Oracle Clusterware and Oracle RAC platform-specific installation and configuration guide in HTML and PDF formats. These installation books contain the preinstallation, installation, and postinstallation information for the various UNIX, Linux, and Windows platforms on which Oracle Clusterware and Oracle RAC operate.

- *Oracle Database 2 Day + Real Application Clusters Guide*

  This task-oriented guide helps you understand the basic steps required to install, configure, and administer Oracle Clusterware and Oracle Real Application Clusters on a two-node system using Oracle Linux 5.0.

- *Oracle Real Application Clusters Administration and Deployment Guide*

  This is an essential companion book that describes topics including instance management, tuning, backup and recovery, and managing services.

- *Oracle Database 2 Day DBA*
- *Oracle Database Administrator's Guide*
- *Oracle Database Net Services Administrator's Guide*
- *Oracle Database Platform Guide for Microsoft Windows*
- *Oracle Database Administrator's Reference for Linux and UNIX-Based Operating Systems*
- *Oracle Database Error Messages*

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Changes in This Release for Oracle Clusterware Administration and Deployment Guide

This preface contains:

- Changes in Oracle Clusterware 12c Release 1 (12.1)

Changes in Oracle Clusterware 12*c* Release 1 (12.1)

The following are changes in *Oracle Clusterware Administration and Deployment Guide* for Oracle Clusterware 12*c*:

- Changes in Oracle Clusterware 12c Release 1 (12.1.0.2)
- Changes in Oracle Clusterware 12c Release 1 (12.1.0.1)
- Deprecated Features
- Desupported Features

Changes in Oracle Clusterware 12*c* Release 1 (12.1.0.2)

The following features are new in this release:

> **See Also:** *Oracle Database New Features Guide* for a complete description of the features in Oracle Database 12*c*

- **Rapid Home Provisioning**

  Rapid Home Provisioning enables you to deploy Oracle homes based on images stored in a catalog of precreated software homes.

  > **See Also:** Chapter 5, "Rapid Home Provisioning" for more information

- **Memory Guard Does Not Require Oracle Database QoS Management to be Active**

  With this release, Memory Guard is enabled by default independent of whether you use Oracle Database Quality of Service Management (Oracle Database QoS Management). Memory Guard detects memory stress on a node and causes new sessions to be directed to other instances until the existing workload drains and frees memory. When free memory increases on the node, then services are enabled again to automatically accept new connections.

- **Oracle Clusterware support for the Diagnosability Framework**

  The Diagnosability Framework enables Oracle products to use a standardized and simplified way of storing and analyzing diagnosability data.

  > **See Also:** "Oracle Clusterware Diagnostic and Alert Log Data" on page J-14 for more information

- **Oracle Trace File Analyzer Collector**

  The Oracle Trace File Analyzer (TFA) Collector is a diagnostic collection utility to simplify diagnostic data collection for Oracle Clusterware, Oracle Grid Infrastructure, and Oracle Real Application Clusters (Oracle RAC) systems.

  > **See Also:** "Oracle Trace File Analyzer Collector" on page J-20 for more information

- **Automatic Installation of Grid Infrastructure Management Repository**

  The Grid Infrastructure Management Repository is automatically installed with Oracle Grid Infrastructure 12*c* release 1 (12.1.0.2). The Grid Infrastructure Management Repository enables such features as Cluster Health Monitor, Oracle Database QoS Management, and Rapid Home Provisioning, and provides a historical metric repository that simplifies viewing of past performance and diagnosis of issues. This capability is fully integrated into Oracle Enterprise Manager Cloud Control for seamless management.

Changes in Oracle Clusterware 12*c* Release 1 (12.1.0.1)

The following features are new in this release:

> **See Also:** *Oracle Database New Features Guide* for a complete description of the features in Oracle Database 12*c*

- **Cluster Health Monitor Enhancements for Oracle Flex Clusters**

  Cluster Health Monitor (CHM) has been enhanced to provide a highly available server monitor service that provides improved detection and analysis of operating system and cluster resource-related degradation and failures. In addition, CHM supports Oracle Flex Clusters configurations, including the ability for data collectors to collect from every node of the cluster and provide a single cluster representation of the data.

  > **See Also:** "Cluster Health Monitor" on page J-3 for more information

- **Cluster Resources for Oracle ACFS and Oracle ADVM**

  Oracle Clusterware resource support includes enhancements for Oracle homes stored on Oracle Automatic Storage Management Cluster File System (Oracle ACFS), Oracle ACFS General Purpose file systems for Grid homes, and Oracle ASM Dynamic Volume Manager (Oracle ADVM) volumes. These resources, managed by Oracle Clusterware, support automatic loading of Oracle ACFS, Oracle ADVM and OKS drivers, disk group mounts, dynamic volume enablement, and automatic Oracle ACFS file system mounts.

  > **See Also:** *Oracle Automatic Storage Management Administrator's Guide* for more information

- **Oracle Flex Clusters**

  Oracle Flex Clusters is a new concept, which joins together a traditional closely-coupled cluster with a modest node count with a large number of loosely-coupled nodes. To support various configurations that can be established using this new concept, SRVCTL provides new commands and command options to ease the installation and configuration.

  > **See Also:** Chapter 4, "Oracle Flex Clusters" for more information

- **IPv6 Support for Public Networks**

  Oracle Clusterware 12*c* supports IPv6-based public IP and VIP addresses.

  IPv6-based IP addresses have become the latest standard for the information technology infrastructure in today's data centers. With this release, Oracle RAC and Oracle Grid Infrastructure support this standard. You can configure cluster nodes during installation with either IPv4 or IPv6 addresses on the same network. Database clients can connect to either IPv4 or IPv6 addresses. The Single Client Access Name (SCAN) listener automatically redirects client connects to the appropriate database listener for the IP protocol of the client request.

  > **See Also:** "Understanding Network Address Configuration Requirements" on page 2-11

- **Shared Grid Naming Service**

  One instance of Grid Naming Service (GNS) can provide name resolution service for any number of clusters.

  > **See Also:** "Oracle Clusterware Network Configuration Concepts" on page 1-5 for more information

- **Oracle Grid Infrastructure User Support on Windows**

  Starting with Oracle Database 12*c*, Oracle Database supports the use of an Oracle Grid Infrastructure installation owner account (Grid user) to own the Oracle Grid Infrastructure home, which you can specify during installation time. The Grid user can be either a built-in account or a Windows user account. A Windows Grid user account should be a low-privileged (non-Administrator) account, so that the Grid user has a limited set of privileges. Installing Oracle Grid Infrastructure using a Grid user account helps to ensure that Oracle Database services have only those privileges required to run Oracle products.

  In prior releases on Windows operating systems, Oracle services ran as local system privileges which are fully privileged. This feature allows the services to run with user privileges to allow tighter control of security. Oracle creates the Windows database service for the Oracle Grid Infrastructure management repository under the security context of the Windows user specified when installing the Oracle Grid Infrastructure, which is the Grid user referred to in the previous paragraph.

  In support of this feature, Oracle enhanced some Cluster Verification Utility (CVU) commands and added CRSCTL commands for managing wallets.

  > **See Also:** *Oracle Database Platform Guide for Microsoft Windows*

- **Oracle Grid Infrastructure Rolling Migration for One-Off Patches**

Oracle Grid Infrastructure one-off patch rolling migration and upgrade for Oracle Automatic Storage Management (Oracle ASM) and Oracle Clusterware enables you to independently upgrade or patch clustered Oracle Grid Infrastructure nodes with one-off patches, without affecting database availability. This feature provides greater uptime and patching flexibility. This release also introduces a new Cluster state, "Rolling Patch." Operations allowed in a patch quiesce state are similar to the existing "Rolling Upgrade" cluster state.

- **Policy-Based Cluster Management and Administration**

  Oracle Grid Infrastructure allows running multiple applications in one cluster. Using a policy-based approach, the workload introduced by these applications can be allocated across the cluster using a **cluster configuration policy**. In addition, a **cluster configuration policy set** enables different cluster configuration policies to be applied to the cluster over time as required. You can define cluster configuration policies using a web-based interface or a command-line interface.

  Hosting various workloads in the same cluster helps to consolidate the workloads into a shared infrastructure that provides high availability and scalability. Using a centralized policy-based approach allows for dynamic resource reallocation and prioritization as the demand changes.

  > **See Also:** "Overview of Cluster Configuration Policies and the Policy Set" on page 3-9 for more information

  Oracle Clusterware 12*c* includes a `generic_application` resource type for fast integration and high availability for any type of application.

  > **See Also:** "Resource Types" on page 9-2 for more information about the `generic_application` resource type

- **Oracle Flex ASM**

  Oracle Flex ASM enables the Oracle ASM instance to run on a separate physical server from the database servers. Any number of Oracle ASM servers can be clustered to support much larger database clients, thereby eliminating a single point of failure.

  > **See Also:** *Oracle Automatic Storage Management Administrator's Guide* for more information about Oracle Flex ASM

- **Restricting Service Registration with Valid Node Checking**

  A standalone Oracle Database listener restricts clients from accessing a database registered with this listener using various conditions, such as the subnet, from which these clients are connecting. Listeners that Oracle Grid Infrastructure manages can now be configured, accordingly.

- **What-If Command Evaluation**

  Oracle Clusterware 12*c* provides a set of evaluation commands and APIs to determine the impact of a certain operation before the respective operation is actually executed.

  > **See Also:**
  >
  > - "Using the eval Command" on page E-3 for more information about the evaluation commands
  > - "What-If APIs" on page H-12 for more information about the APIs

- **Online Resource Attribute Modification**

  Oracle Clusterware manages hardware and software components for high availability using a resource model. You use resource attributes to define how Oracle Clusterware manages those resources. You can modify certain resource attributes and implement those changes without having to restart the resource using online resource attribute modification. You manage online resource attribute modification with certain SRVCTL and CRSCTL commands.

- **Oracle Cluster Registry Backup in Oracle ASM Disk Group Support**

  The Oracle Cluster Registry (OCR) backup mechanism enables storing the OCR backup in an Oracle ASM disk group. Storing the OCR backup in an Oracle ASM disk group simplifies OCR management by permitting access to the OCR backup from any node in the cluster should an OCR recovery become necessary.

  > **See Also:** "Managing Oracle Cluster Registry and Oracle Local Registry" on page 6-2

Deprecated Features

The following features are deprecated with this release, and may be desupported in a future release:

- **Deprecation of Oracle Restart**

  Oracle Restart is deprecated in Oracle Database 12*c*. Oracle Restart is currently restricted to manage single-instance Oracle databases and Oracle ASM instances only, and is subject to desupport in future releases. Oracle continues to provide Oracle ASM as part of the Oracle Grid Infrastructure installation for Standalone and Cluster deployments.

- **Management of Cluster Administrators using a stored list of administrative users**

  This method of administration is being replaced with more comprehensive management of administrative user roles by configuring the access control list of the policy set.

- **Deprecation of single-letter SRVCTL CLI options**

  All SRVCTL commands have been enhanced to accept full-word options instead of the single-letter options. All new SRVCTL command options added in this release support full-word options, only, and do not have single-letter equivalents. This feature might be desupported in a future release.

Desupported Features

> **See Also:** *Oracle Database Upgrade Guide*

- **Oracle Cluster File System on Windows**

  Oracle no longer supports Oracle Cluster File System (OCFS) on Windows.

- **Oracle Enterprise Manager Database Control**

  Oracle Enterprise Manager Database Control is replaced by Oracle Enterprise Manager Database Express.

- **Raw (block) storage devices for Oracle Database and related technologies**

  Oracle Database 12*c* release 1 (12.1) and its related grid technologies, such as Oracle Clusterware, no longer support the direct use of raw or block storage devices. You must move existing files from raw or block devices to Oracle ASM before you upgrade to Oracle Clusterware 12*c* release 1 (12.1).

1

Introduction to Oracle Clusterware

Oracle Clusterware enables servers to communicate with each other, so that they appear to function as a collective unit. This combination of servers is commonly known as a cluster. Although the servers are standalone servers, each server has additional processes that communicate with other servers. In this way the separate servers appear as if they are one system to applications and end users.

This chapter includes the following topics:

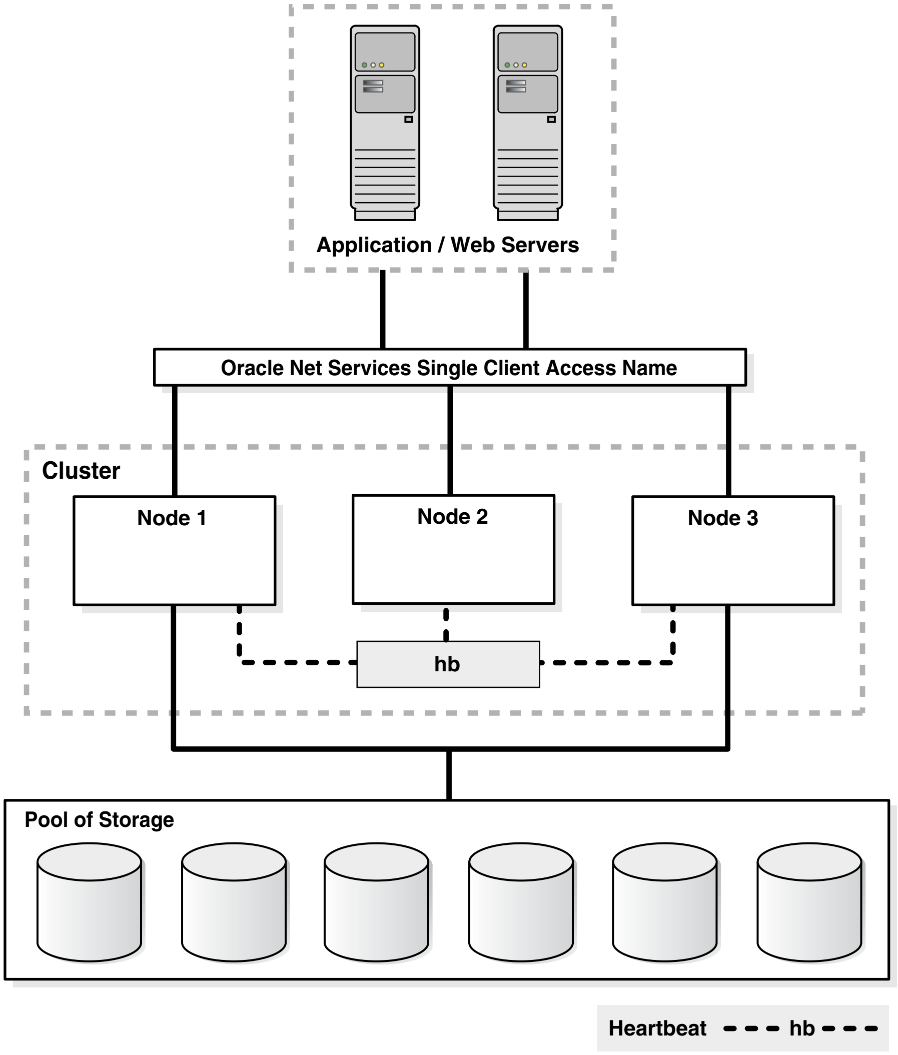
What is Oracle Clusterware?

Oracle Clusterware provides the infrastructure necessary to run Oracle Real Application Clusters (Oracle RAC). Oracle Clusterware also manages **resource**s, such as virtual IP (VIP) addresses, databases, listeners, services, and so on. These resources are generally named `ora.`*`entity_name`*`.`*`resource_type_abbreviation`*, such as `ora.mydb.db`, which is the name of a resource that is a database. (Some examples of *`abbreviation`* are `db` for database, `lsnr` for listener, and `vip` for VIP.) Oracle does not support editing these resources except under the explicit direction of My Oracle Support.

> **See Also:** Chapter 2, "Administering Oracle Clusterware" and Chapter 9, "Making Applications Highly Available Using Oracle Clusterware" for more information

Figure 1–1 shows a configuration that uses Oracle Clusterware to extend the basic single-instance Oracle Database architecture. In Figure 1–1, the cluster is running Oracle Database and is actively servicing applications and users. Using Oracle Clusterware, you can use the same high availability mechanisms to make your Oracle database and your custom applications highly available.

Figure 1–1 Oracle Clusterware Configuration


Application / Web Servers
Oracle Net Services Single Client Access Name
Cluster
Node 1
Node 2
Node 3
hb
Pool of Storage
Heartbeat hb

The benefits of using a cluster include:

- Scalability of applications
- Reduce total cost of ownership for the infrastructure by providing a scalable system with low-cost commodity hardware
- Ability to fail over
- Increase throughput on demand for cluster-aware applications, by adding servers to a cluster to increase cluster resources
- Increase throughput for cluster-aware applications by enabling the applications to run on all of the nodes in a cluster
- Ability to program the startup of applications in a planned order that ensures dependent processes are started in the correct sequence
- Ability to monitor processes and restart them if they stop
- Eliminate unplanned downtime due to hardware or software malfunctions
- Reduce or eliminate planned downtime for software maintenance

You can program Oracle Clusterware to manage the availability of user applications and Oracle databases. In an Oracle RAC environment, Oracle Clusterware manages all of the resources automatically. All of the applications and processes that Oracle Clusterware manages are either **cluster resource**s or **local resource**s.

Oracle Clusterware is required for using Oracle RAC; it is the only clusterware that you need for platforms on which Oracle RAC operates. Although Oracle RAC continues to support many third-party clusterware products on specific platforms, you must also install and use Oracle Clusterware. Note that the servers on which you want to install and run Oracle Clusterware must use the same operating system.

Using Oracle Clusterware eliminates the need for proprietary vendor clusterware and provides the benefit of using only Oracle software. Oracle provides an entire software solution, including everything from disk management with Oracle Automatic Storage Management (Oracle ASM) to data management with Oracle Database and Oracle RAC. In addition, Oracle Database features, such as Oracle Services, provide advanced functionality when used with the underlying Oracle Clusterware high availability framework.

Oracle Clusterware has two stored components, besides the binaries: The voting files, which record node membership information, and the Oracle Cluster Registry (OCR), which records cluster configuration information. Voting files and OCRs must reside on shared storage available to all cluster member nodes.

Understanding System Requirements for Oracle Clusterware

To use Oracle Clusterware, you must understand the hardware and software concepts and requirements as described in the following sections:

- Oracle Clusterware Hardware Concepts and Requirements
- Oracle Clusterware Operating System Concepts and Requirements
- Oracle Clusterware Software Concepts and Requirements
- Oracle Clusterware Network Configuration Concepts
- Overview of Upgrading and Patching Oracle Clusterware

Oracle Clusterware Hardware Concepts and Requirements

> **Note:** Many hardware providers have validated cluster configurations that provide a single part number for a cluster. If you are new to clustering, then use the information in this section to simplify your hardware procurement efforts when you purchase hardware to create a cluster.

A cluster consists of one or more servers. Access to an external network is the same for a server in a cluster (also known as a *cluster member* or *node*) as for a standalone server. However, a server that is part of a cluster, otherwise known as a node or a cluster member, requires a second network. This second network is referred to as the interconnect. For this reason, cluster member nodes require at least two network interface cards: one for a public network and one for a private network. The interconnect network is a private network using a switch (or multiple switches) that only the nodes in the cluster can access.[1]

Note: Oracle does not support using crossover cables as Oracle Clusterware interconnects.

Cluster size is determined by the requirements of the workload running on the cluster and the number of nodes that you have configured in the cluster. If you are implementing a cluster for high availability, then configure redundancy for all of the components of the infrastructure as follows:

- At least two network interfaces for the public network, bonded to provide one address
- At least two network interfaces for the private interconnect network

The cluster requires cluster-aware storage[2] that is connected to each server in the cluster. This may also be referred to as a multihost device. Oracle Clusterware supports Network File Systems (NFSs), iSCSI, Direct Attached Storage (DAS), Storage Area Network (SAN) storage, and Network Attached Storage (NAS).

To provide redundancy for storage, generally provide at least two connections from each server to the cluster-aware storage. There may be more connections depending on your I/O requirements. It is important to consider the I/O requirements of the entire cluster when choosing your storage subsystem.

Most servers have at least one local disk that is internal to the server. Often, this disk is used for the operating system binaries; you can also use this disk for the Oracle software binaries. The benefit of each server having its own copy of the Oracle binaries is that it increases high availability, so that corruption of one binary does not affect all of the nodes in the cluster simultaneously. It also allows rolling upgrades, which reduce downtime.

Oracle Clusterware Operating System Concepts and Requirements

Each server must have an operating system that is certified with the Oracle Clusterware version you are installing. Refer to the certification matrices available in the *Oracle Grid Infrastructure Installation Guide* for your platform or on My Oracle Support (formerly Oracle*MetaLink*) for details, which are available from the following URL:

http://www.oracle.com/technetwork/database/clustering/tech-generic-unix-new-166583.html

When the operating system is installed and working, you can then install Oracle Clusterware to create the cluster. Oracle Clusterware is installed independently of Oracle Database. After you install Oracle Clusterware, you can then install Oracle Database or Oracle RAC on any of the nodes in the cluster.

See Also: *Oracle Grid Infrastructure Installation Guide*

Oracle Clusterware Software Concepts and Requirements

Oracle Clusterware uses voting files to provide fencing and cluster node membership determination. OCR provides cluster configuration information. You can place the Oracle Clusterware files on either Oracle ASM or on shared common disk storage. If

[1] Oracle Clusterware supports up to 100 nodes in a cluster on configurations running Oracle Database 10*g* release 2 (10.2) and later releases.

[2] Cluster-aware storage may also be referred to as a multihost device.

you configure Oracle Clusterware on storage that does not provide file redundancy, then Oracle recommends that you configure multiple locations for OCR and voting files. The voting files and OCR are described as follows:

- **Voting Files**

  Oracle Clusterware uses voting files to determine which nodes are members of a cluster. You can configure voting files on Oracle ASM, or you can configure voting files on shared storage.

  If you configure voting files on Oracle ASM, then you do not need to manually configure the voting files. Depending on the redundancy of your disk group, an appropriate number of voting files are created.

  If you do not configure voting files on Oracle ASM, then for high availability, Oracle recommends that you have a minimum of three voting files on physically separate storage. This avoids having a single point of failure. If you configure a single voting file, then you must use external mirroring to provide redundancy.

  Oracle recommends that you do not use more than five voting files, even though Oracle supports a maximum number of 15 voting files.

- **Oracle Cluster Registry**

  Oracle Clusterware uses the Oracle Cluster Registry (OCR) to store and manage information about the components that Oracle Clusterware controls, such as Oracle RAC databases, listeners, virtual IP addresses (VIPs), and services and any applications. OCR stores configuration information in a series of key-value pairs in a tree structure. To ensure cluster high availability, Oracle recommends that you define multiple OCR locations. In addition:

  - You can have up to five OCR locations
  - Each OCR location must reside on shared storage that is accessible by all of the nodes in the cluster
  - You can replace a failed OCR location online if it is not the only OCR location
  - You must update OCR through supported utilities such as Oracle Enterprise Manager, the Oracle Clusterware Control Utility (CRSCTL), the Server Control Utility (SRVCTL), the OCR configuration utility (OCRCONFIG), or the Database Configuration Assistant (DBCA)

    See Also: Chapter 2, "Administering Oracle Clusterware" for more information about voting files and OCR

Oracle Clusterware Network Configuration Concepts

Oracle Clusterware enables a dynamic Oracle Grid Infrastructure through the self-management of the network requirements for the cluster. Oracle Clusterware 12*c* supports the use of **Dynamic Host Configuration Protocol (DHCP)** or stateless address autoconfiguration for the VIP addresses and the **Single Client Access Name (SCAN)** address, but not the public address. DHCP provides dynamic assignment of IPv4 VIP addresses, while Stateless Address Autoconfiguration provides dynamic assignment of IPv6 VIP addresses.

When you are using Oracle RAC, all of the clients must be able to reach the database, which means that the clients must resolve VIP and SCAN names to all of the VIP and SCAN addresses, respectively. This problem is solved by the addition of **Grid Naming Service (GNS)** to the cluster. GNS is linked to the corporate Domain Name Service (DNS) so that clients can resolve host names to these dynamic addresses and transparently connect to the cluster and the databases. Oracle supports using GNS without DHCP or zone delegation in Oracle Clusterware 12*c* (as with Oracle Flex ASM server clusters, which you can configure without zone delegation or dynamic networks).

Note: Oracle *does not* support using GNS without DHCP or zone delegation on Windows.

See Also: *Oracle Automatic Storage Management Administrator's Guide* for more information about Oracle Flex ASM

Beginning with Oracle Clusterware 12*c*, a GNS instance can now service multiple clusters rather than just one, thus only a single domain must be delegated to GNS in DNS. GNS still provides the same services as in previous versions of Oracle Clusterware.

The cluster in which the GNS server runs is referred to as the **server cluster**. A **client cluster** advertises its names with the server cluster. Only one GNS daemon process can run on the server cluster. Oracle Clusterware puts the GNS daemon process on one of the nodes in the cluster to maintain availability.

In previous, single-cluster versions of GNS, the single cluster could easily locate the GNS service provider within itself. In the multicluster environment, however, the client clusters must know the GNS address of the server cluster. Given that address, client clusters can find the GNS server running on the server cluster.

In order for GNS to function on the server cluster, you must have the following:

- The DNS administrator must delegate a zone for use by GNS
- A GNS instance must be running somewhere on the network and it must not be blocked by a firewall
- All of the node names in a set of clusters served by GNS must be unique

See Also: "Overview of Grid Naming Service" on page 2-3 for information about administering GNS

Single Client Access Name (SCAN)

The SCAN is a domain name registered to at least one and up to three IP addresses, either in DNS or GNS. When using GNS and DHCP, Oracle Clusterware configures the VIP addresses for the SCAN name that is provided during cluster configuration.

The node VIP and the three SCAN VIPs are obtained from the DHCP server when using GNS. If a new server joins the cluster, then Oracle Clusterware dynamically obtains the required VIP address from the DHCP server, updates the cluster resource, and makes the server accessible through GNS.

Note: "Understanding SCAN Addresses and Client Service Connections" on page 2-13 for more information about SCAN

Configuring Addresses Manually

Alternatively, you can choose manual address configuration, in which you configure the following:

- One public address and host name for each node.

- One VIP address for each node.

  You must assign a VIP address to each node in the cluster. Each VIP address must be on the same subnet as the public IP address for the node and should be an address that is assigned a name in the DNS. Each VIP address must also be unused and unpingable from within the network before you install Oracle Clusterware.

- Up to three SCAN addresses for the entire cluster.

Note: The SCAN must resolve to at least one address on the public network. For high availability and scalability, Oracle recommends that you configure the SCAN to resolve to three addresses on the public network.

See Also: Your platform-specific *Oracle Grid Infrastructure Installation Guide* installation documentation for information about system requirements and configuring network addresses

Overview of Oracle Clusterware Platform-Specific Software Components

When Oracle Clusterware is operational, several platform-specific processes or services run on each node in the cluster. This section describes these various processes and services.

The Oracle Clusterware Technology Stack

Oracle Clusterware consists of two separate technology stacks: an upper technology stack anchored by the Cluster Ready Services (CRS) daemon (CRSD) and a lower technology stack anchored by the Oracle High Availability Services daemon (OHASD). These two technology stacks have several processes that facilitate cluster operations. The following sections describe these technology stacks in more detail:

- The Cluster Ready Services Technology Stack
- The Oracle High Availability Services Technology Stack

The Cluster Ready Services Technology Stack

The following list describes the processes that comprise CRS:

- **Cluster Ready Services (CRS)**: The primary program for managing high availability operations in a cluster.

  The CRSD manages cluster resources based on the configuration information that is stored in OCR for each resource. This includes start, stop, monitor, and failover operations. The CRSD process generates events when the status of a resource changes. When you have Oracle RAC installed, the CRSD process monitors the Oracle database instance, listener, and so on, and automatically restarts these components when a failure occurs.

- **Cluster Synchronization Services (CSS)**: Manages the cluster configuration by controlling which nodes are members of the cluster and by notifying members when a node joins or leaves the cluster. If you are using certified third-party clusterware, then CSS processes interface with your clusterware to manage node membership information.

The `cssdagent` process monitors the cluster and provides I/O fencing. This service formerly was provided by Oracle Process Monitor Daemon (`oprocd`), also known as `OraFenceService` on Windows. A `cssdagent` failure may result in Oracle Clusterware restarting the node.

- **Oracle ASM**: Provides disk management for Oracle Clusterware and Oracle Database.
- **Cluster Time Synchronization Service (CTSS)**: Provides time management in a cluster for Oracle Clusterware.
- **Event Management (EVM)**: A background process that publishes events that Oracle Clusterware creates.
- **Grid Naming Service (GNS)**: Handles requests sent by external DNS servers, performing name resolution for names defined by the cluster.
- **Oracle Agent (`oraagent`)**: Extends clusterware to support Oracle-specific requirements and complex resources. This process runs server callout scripts when FAN events occur. This process was known as RACG in Oracle Clusterware 11*g* release 1 (11.1).
- **Oracle Notification Service (ONS)**: A publish and subscribe service for communicating Fast Application Notification (FAN) events.
- **Oracle Root Agent(`orarootagent`)**: A specialized `oraagent` process that helps the CRSD manage resources owned by `root`, such as the network, and the Grid virtual IP address.

The Cluster Synchronization Service (CSS), Event Management (EVM), and Oracle Notification Services (ONS) components communicate with other cluster component layers on other nodes in the same cluster database environment. These components are also the main communication links between Oracle Database, applications, and the Oracle Clusterware high availability components. In addition, these background processes monitor and manage database operations.

The Oracle High Availability Services Technology Stack

The following list describes the processes that comprise the Oracle High Availability Services technology stack:

- **`appagent`**: Protects any resources of the `application` resource type used in previous versions of Oracle Clusterware.

  See Also: "Resources" on page 9-2 for more information about `appagent`

- **Cluster Logger Service (`ologgerd`)**: Receives information from all the nodes in the cluster and persists in an Oracle Grid Infrastructure Management Repository-based database. This service runs on only two nodes in a cluster.
- **Grid Interprocess Communication (GIPC)**: A support daemon that enables Redundant Interconnect Usage.
- **Grid Plug and Play (GPNPD)**: Provides access to the Grid Plug and Play profile, and coordinates updates to the profile among the nodes of the cluster to ensure that all of the nodes have the most recent profile.
- **Multicast Domain Name Service (mDNS)**: Used by Grid Plug and Play to locate profiles in the cluster, and by GNS to perform name resolution. The mDNS process is a background process on Linux and UNIX and on Windows.

- **Oracle Agent (`oraagent`)**: Extends clusterware to support Oracle-specific requirements and complex resources. This process manages daemons that run as the Oracle Clusterware owner, like the GIPC, GPNPD, and GIPC daemons.

  Note: This process is distinctly different from the process of the same name that runs in the Cluster Ready Services technology stack.

- **Oracle Root Agent (`orarootagent`)**: A specialized `oraagent` process that helps the CRSD manage resources owned by `root`, such as the Cluster Health Monitor (CHM).

  Note: This process is distinctly different from the process of the same name that runs in the Cluster Ready Services technology stack.

  See Also: "Overview of Managing Oracle Clusterware Environments" on page 1-13 for more information about CHM

- **`scriptagent`**: Protects resources of resource types other than `application` when using shell or batch scripts to protect an application.

  See Also: "Resources" on page 9-2 for more information about `scriptagent`

- **System Monitor Service (`osysmond`)**: The monitoring and operating system metric collection service that sends the data to the cluster logger service. This service runs on every node in a cluster.

Table 1–1 lists the processes and services associated with Oracle Clusterware components. In Table 1–1, if a UNIX or a Linux system process has an (r) beside it, then the process runs as the `root` user.

Table 1–1 List of Processes and Services Associated with Oracle Clusterware Components[1]

Oracle Clusterware Component	Linux/UNIX Process	Windows Processes
Oracle ASM[2]		
CRS	`crsd.bin` (r)	`crsd.exe`
CSS	`ocssd.bin`, `cssdmonitor`, `cssdagent`	`cssdagent.exe`, `cssdmonitor.exe` `ocssd.exe`
CTSS	`octssd.bin` (r)	`octssd.exe`
EVM	`evmd.bin`, `evmlogger.bin`	`evmd.exe`
GIPC	`gipcd.bin`	
GNS	`gnsd` (r)	`gnsd.exe`
Grid Plug and Play	`gpnpd.bin`	`gpnpd.exe`
LOGGER	`ologgerd.bin` (r)	`ologgerd.exe`
Master Diskmon	`diskmon.bin`	
mDNS	`mdnsd.bin`	`mDNSResponder.exe`

Table 1–1 (Cont.) List of Processes and Services Associated with Oracle Clusterware Components[1]

Oracle Clusterware Component	Linux/UNIX Process	Windows Processes
Oracle agent	`oraagent.bin` (Oracle Clusterware 12*c* release 1 (12.1) and 11*g* release 2 (11.2)), or `racgmain` and `racgimon` (Oracle Clusterware 11*g* release 1 (11.1))	`oraagent.exe`
Oracle High Availability Services	`ohasd.bin` (r)	`ohasd.exe`
ONS	`ons`	`ons.exe`
Oracle root agent	`orarootagent` (r)	`orarootagent.exe`
SYSMON	`osysmond.bin` (r)	`osysmond.exe`

[1] The only Windows services associated with the Oracle Grid Infrastructure are OracleOHService (OHASD), Oracle ASM, listener services (including node listeners and SCAN listeners), and management database. Oracle ASM can be considered part of the Oracle Clusterware technology stack when OCR is stored on Oracle ASM. The listeners and management database are Oracle Clusterware resources and are not properly part of the Oracle Clusterware technology stack.

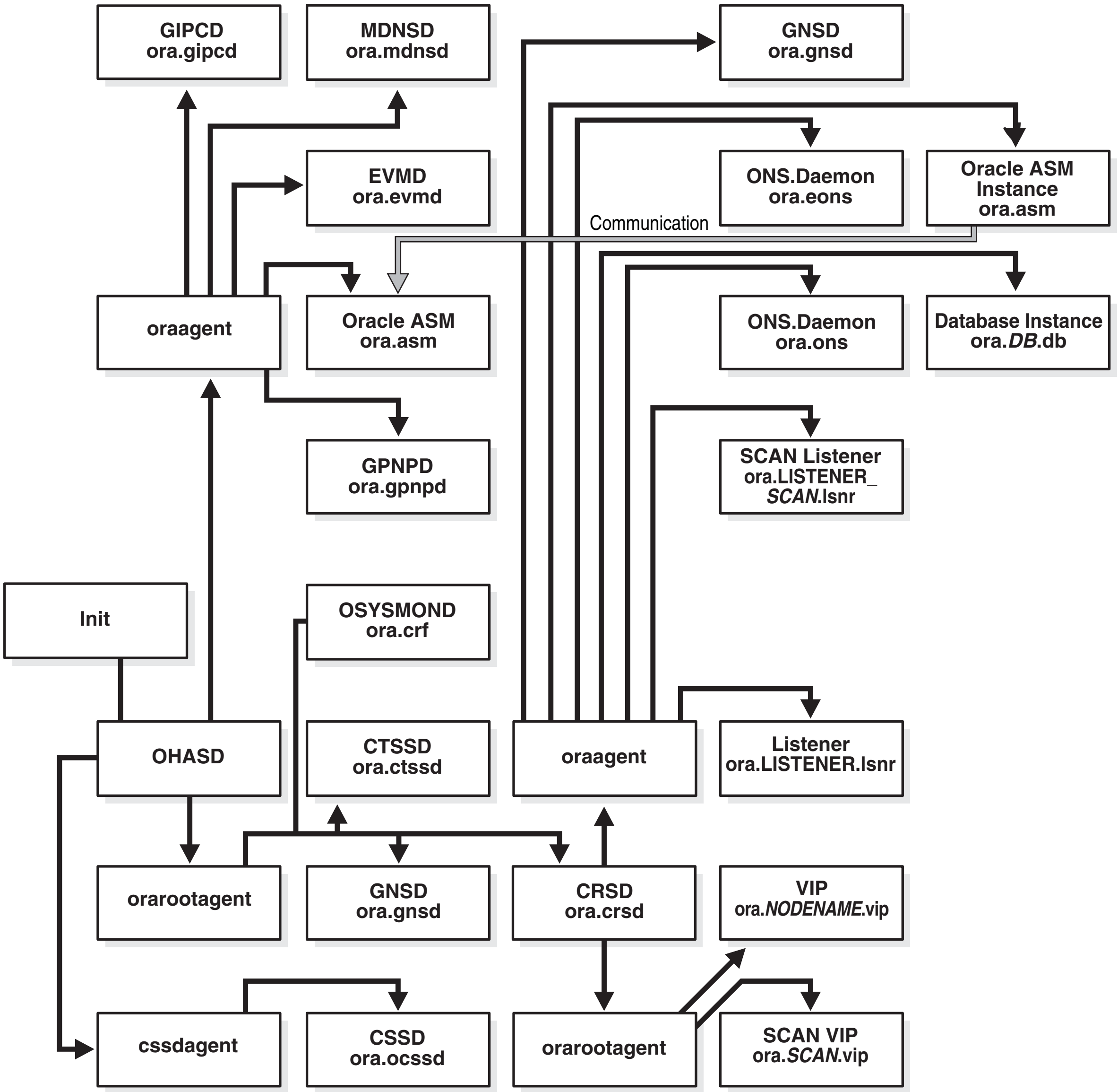
[2] Oracle ASM is not just one process, but an instance. Given Oracle Flex ASM, Oracle ASM does not necessarily run on every cluster node but only some of them.

See Also: "Oracle Clusterware Diagnostic and Alert Log Data" on page J-14 for information about the location of log files created for processes

Note: Oracle Clusterware on Linux platforms can have multiple threads that appear as separate processes with unique process identifiers.

Figure 1–2 illustrates cluster startup.

Figure 1–2 Cluster Startup



Oracle Clusterware Processes on Windows Systems

Oracle Clusterware processes on Microsoft Windows systems include the following:

- `mDNSResponder.exe`: Manages name resolution and service discovery within attached subnets
- `OracleOHService`: Starts all of the Oracle Clusterware daemons

Overview of Installing Oracle Clusterware

The following section introduces the installation processes for Oracle Clusterware.

Note: Install Oracle Clusterware with the Oracle Universal Installer.

Oracle Clusterware Version Compatibility

You can install different releases of Oracle Clusterware, Oracle ASM, and Oracle Database on your cluster. Follow these guidelines when installing different releases of software on your cluster:

- You can only have one installation of Oracle Clusterware running in a cluster, and it must be installed into its own home (`Grid_home`). The release of Oracle Clusterware that you use must be equal to or higher than the Oracle ASM and Oracle RAC versions that run in the cluster. You cannot install a version of Oracle RAC that was released after the version of Oracle Clusterware that you run on the cluster. In other words:
  - Oracle Clusterware 12*c* supports Oracle ASM 12*c* only, because Oracle ASM is in the Oracle Grid Infrastructure home, which also includes Oracle Clusterware
  - Oracle Clusterware 12*c* supports Oracle Database 12*c*, Oracle Database 11*g* release 2 (11.2) and 11*g* release 1 (11.1), and Oracle Database 10*g* release 2 (10.2) and 10*g* release 1 (10.1)
  - Oracle ASM 12*c* requires Oracle Clusterware 12*c* and supports Oracle Database 12*c*, Oracle Database 11*g* release 2 (11.2), Oracle Database 11*g* release 1 (11.1), Oracle Database 10*g* release 2 (10.2), and 10*g* release 1 (10.1)
  - Oracle Database 12*c* requires Oracle Clusterware 12*c*

    For example:

    - If you have Oracle Clusterware 12*c* installed as your clusterware, then you can have an Oracle Database 10*g* release 1 (10.1) single-instance database running on one node, and separate Oracle Real Application Clusters 10*g* release 1 (10.1), 10*g* release 2 (10.2), and Oracle Real Application Clusters 11*g* release 1 (11.1) databases also running on the cluster. However, you cannot have Oracle Clusterware 10*g* release 2 (10.2) installed on your cluster, and install Oracle Real Application Clusters 11*g*. You can install Oracle Database 11*g* single-instance on a node in an Oracle Clusterware 10*g* release 2 (10.2) cluster.
    - When using different Oracle ASM and Oracle Database releases, the functionality of each depends on the functionality of the earlier software release. Thus, if you install Oracle Clusterware 11*g* and you later configure Oracle ASM, and you use Oracle Clusterware to support an existing Oracle Database 10*g* release 2 (10.2.0.3) installation, then the Oracle ASM functionality is equivalent only to that available in the 10*g* release 2 (10.2.0.3) release version. Set the compatible attributes of a disk group to the appropriate release of software in use.

    See Also: *Oracle Automatic Storage Management Administrator's Guide* for information about compatible attributes of disk groups

- There can be multiple Oracle homes for the Oracle database (both single instance and Oracle RAC) in the cluster. The Oracle homes for all nodes of an Oracle RAC database must be the same.
- You can use different users for the Oracle Clusterware and Oracle database homes if they belong to the same primary group.
- As of Oracle Clusterware 12*c*, there can only be one installation of Oracle ASM running in a cluster. Oracle ASM is always the same version as Oracle

Clusterware, which must be the same (or higher) release than that of the Oracle database.

- For Oracle RAC running Oracle9*i* you must run an Oracle9*i* cluster. For UNIX systems, that is HACMP, Serviceguard, Sun Cluster, or Veritas SF. For Windows and Linux systems, that is the Oracle Cluster Manager. To install Oracle RAC 10*g*, you must also install Oracle Clusterware.
- Oracle recommends that you do not run different cluster software on the same servers unless they are certified to work together. However, if you are adding Oracle RAC to servers that are part of a cluster, either migrate to Oracle Clusterware or ensure that:
  - The clusterware you run is supported to run with Oracle RAC 12*c*.
  - You have installed the correct options for Oracle Clusterware and the other vendor clusterware to work together.

See Also: *Oracle Grid Infrastructure Installation Guide* for more version compatibility information

Overview of Upgrading and Patching Oracle Clusterware

Oracle supports out-of-place upgrades, only, because Oracle Clusterware 12*c* must have its own, new Grid home. For Oracle Clusterware 12*c*, Oracle supports in-place or out-of-place patching. Oracle supports patch bundles and one-off patches for in-place patching but only supports patch sets and major point releases for out-of-place upgrades.

In-place patching replaces the Oracle Clusterware software with the newer version in the same Grid home. Out-of-place upgrade has both versions of the same software present on the nodes at the same time, in different Grid homes, but only one version is active.

Rolling upgrades avoid downtime and ensure continuous availability of Oracle Clusterware while the software is upgraded to the new version. When you upgrade to Oracle Clusterware 12*c*, Oracle Clusterware and Oracle ASM binaries are installed as a single binary called the Oracle Grid Infrastructure. You can upgrade Oracle Clusterware in a rolling manner from Oracle Clusterware 10*g* and Oracle Clusterware 11*g*, however you can only upgrade Oracle ASM in a rolling manner from Oracle Database 11*g* release 1 (11.1).

Oracle supports force upgrades in cases where some nodes of the cluster are down.

See Also: Your platform-specific *Oracle Grid Infrastructure Installation Guide* for procedures on upgrading Oracle Clusterware

Overview of Managing Oracle Clusterware Environments

The following list describes the tools and utilities for managing your Oracle Clusterware environment:

- **Cluster Health Monitor (CHM)**: **Cluster Health Monitor** detects and analyzes operating system and cluster resource-related degradation and failures to provide more details to users for many Oracle Clusterware and Oracle RAC issues, such as node eviction. The tool continuously tracks the operating system resource consumption at the node, process, and device levels. It collects and analyzes the clusterwide data. In real-time mode, when thresholds are met, the tool shows an alert to the user. For root-cause analysis, historical data can be replayed to understand what was happening at the time of failure.

> **See Also:** "Cluster Health Monitor" on page J-3 for more information about CHM

- **Cluster Verification Utility (CVU)**: CVU is a command-line utility that you use to verify a range of cluster and Oracle RAC specific components. Use CVU to verify shared storage devices, networking configurations, system requirements, and Oracle Clusterware, and operating system groups and users.

  Install and use CVU for both preinstallation and postinstallation checks of your cluster environment. CVU is especially useful during preinstallation and during installation of Oracle Clusterware and Oracle RAC components to ensure that your configuration meets the minimum installation requirements. Also use CVU to verify your configuration after completing administrative tasks, such as node additions and node deletions.

  > **See Also:** Your platform-specific Oracle Clusterware and Oracle RAC installation guide for information about how to manually install CVU, and Appendix A, "Cluster Verification Utility Reference" for more information about using CVU

- **Oracle Cluster Registry Configuration Tool (OCRCONFIG)**: OCRCONFIG is a command-line tool for OCR administration. You can also use the OCRCHECK and OCRDUMP utilities to troubleshoot configuration problems that affect OCR.

  > **See Also:** Chapter 2, "Administering Oracle Clusterware" for more information about managing OCR

- **Oracle Clusterware Control (CRSCTL)**: CRSCTL is a command-line tool that you can use to manage Oracle Clusterware. Use CRSCTL for general clusterware management, management of individual resources, configuration policies, and server pools for non-database applications.

  Oracle Clusterware 12*c* introduces cluster-aware commands with which you can perform operations from any node in the cluster on another node in the cluster, or on all nodes in the cluster, depending on the operation.

  You can use `crsctl` commands to monitor cluster resources (`crsctl status resource`) and to monitor and manage servers and server pools other than server pools that have names prefixed with `ora.*`, such as `crsctl status server`, `crsctl status serverpool`, `crsctl modify serverpool`, and `crsctl relocate server`. You can also manage Oracle High Availability Services on the entire cluster (`crsctl start | stop | enable | disable | config crs`), using the optional node-specific arguments `-n` or `-all`. You also can use CRSCTL to manage Oracle Clusterware on individual nodes (`crsctl start | stop | enable | disable | config crs`).

  > **See Also:**
  >
  > - Chapter 2, "Administering Oracle Clusterware" for more information about using `crsctl` commands to manage Oracle Clusterware
  > - Appendix E, "Oracle Clusterware Control (CRSCTL) Utility Reference" for a complete list of CRSCTL commands

- **Oracle Enterprise Manager**: Oracle Enterprise Manager has both the Cloud Control and Grid Control GUI interfaces for managing both single instance and Oracle RAC database environments. It also has GUI interfaces to manage Oracle Clusterware and all components configured in the Oracle Grid Infrastructure installation. Oracle recommends that you use Oracle Enterprise Manager to perform administrative tasks.

  See Also: *Oracle Database 2 Day + Real Application Clusters Guide*, *Oracle Real Application Clusters Administration and Deployment Guide*, and Oracle Enterprise Manager online documentation for more information about administering Oracle Clusterware with Oracle Enterprise Manager

- **Oracle Interface Configuration Tool (OIFCFG)**: OIFCFG is a command-line tool for both single-instance Oracle databases and Oracle RAC environments. Use OIFCFG to allocate and deallocate network interfaces to components. You can also use OIFCFG to direct components to use specific network interfaces and to retrieve component configuration information.

  See Also: Appendix D, "Oracle Interface Configuration Tool (OIFCFG) Command Reference"

- **Server Control (SRVCTL)**: SRVCTL is a command-line interface that you can use to manage Oracle resources, such as databases, services, or listeners in the cluster.

  Note: You can only use SRVCTL to manage server pools that have names prefixed with `ora.*`.

  See Also: *Oracle Real Application Clusters Administration and Deployment Guide* for more information about SRVCTL

Overview of Cloning and Extending Oracle Clusterware in Grid Environments

Cloning nodes is the preferred method of creating new clusters. The cloning process copies Oracle Clusterware software images to other nodes that have similar hardware and software. Use cloning to quickly create several clusters of the same configuration. Before using cloning, you must install an Oracle Clusterware home successfully on at least one node using the instructions in your platform-specific Oracle Clusterware installation guide.

For new installations, or if you must install on only one cluster, Oracle recommends that you use the automated and interactive installation methods, such as Oracle Universal Installer or the Provisioning Pack feature of Oracle Enterprise Manager. These methods perform installation checks to ensure a successful installation. To add or delete Oracle Clusterware from nodes in the cluster, use the `addnode.sh` and `rootcrs.pl` scripts.

See Also:

- Chapter 8, "Cloning Oracle Clusterware" for step-by-step cloning procedures
- Oracle Enterprise Manager online Help system for more information about the Provisioning Pack
- Chapter 7, "Adding and Deleting Cluster Nodes"

Overview of the Oracle Clusterware High Availability Framework and APIs

Oracle Clusterware provides many high availability application programming interfaces called CLSCRS APIs that you use to enable Oracle Clusterware to manage applications or processes that run in a cluster. The CLSCRS APIs enable you to provide high availability for all of your applications.

See Also: Appendix H, "Oracle Clusterware C Application Program Interfaces" for more detailed information about the CLSCRS APIs

You can define a VIP address for an application to enable users to access the application independently of the node in the cluster on which the application is running. This is referred to as the application VIP. You can define multiple application VIPs, with generally one application VIP defined for each application running. The application VIP is related to the application by making it dependent on the application resource defined by Oracle Clusterware.

To maintain high availability, Oracle Clusterware components can respond to status changes to restart applications and processes according to defined high availability rules. You can use the Oracle Clusterware high availability framework by registering your applications with Oracle Clusterware and configuring the clusterware to start, stop, or relocate your application processes. That is, you can make custom applications highly available by using Oracle Clusterware to create profiles that monitor, relocate, and restart your applications.

Overview of Cluster Time Management

The Cluster Time Synchronization Service (CTSS) is installed as part of Oracle Clusterware and runs in *observer* mode if it detects a time synchronization service or a time synchronization service configuration, valid or broken, on the system. For example, if the `etc/ntp.conf` file exists on any node in the cluster, then CTSS runs in observer mode even if no time synchronization software is running.

If CTSS detects that there is no time synchronization service or time synchronization service configuration on *any* node in the cluster, then CTSS goes into *active* mode and takes over time management for the cluster.

If CTSS is running in active mode while another, non-NTP, time synchronization software is running, then you can change CTSS to run in observer mode by creating a file called `etc/ntp.conf`. CTSS puts an entry in the alert log about the change to observer mode.

When nodes join the cluster, if CTSS is in active mode, then it compares the time on those nodes to a reference clock located on one node in the cluster. If there is a discrepancy between the two times and the discrepancy is within a certain stepping limit, then CTSS performs *step time synchronization*, which is to step the time, forward or backward, of the nodes joining the cluster to synchronize them with the reference.

Clocks on nodes in the cluster become desynchronized with the reference clock (a time CTSS uses as a basis and is on the first node started in the cluster) periodically for various reasons. When this happens, CTSS performs *slew time synchronization*, which is to speed up or slow down the system time on the nodes until they are synchronized with the reference system time. In this time synchronization method, CTSS does not adjust time backward, which guarantees monotonic increase of the system time.

When Oracle Clusterware starts, if CTSS is running in active mode and the time discrepancy is outside the stepping limit (the limit is 24 hours), then CTSS generates an alert in the alert log, exits, and Oracle Clusterware startup fails. You must manually adjust the time of the nodes joining the cluster to synchronize with the cluster, after which Oracle Clusterware can start and CTSS can manage the time for the nodes.

When performing slew time synchronization, CTSS never runs time backward to synchronize with the reference clock. CTSS periodically writes alerts to the alert log containing information about how often it adjusts time on nodes to keep them synchronized with the reference clock.

CTSS writes entries to the Oracle Clusterware alert log and `syslog` when it:

- Detects a time change
- Detects significant time difference from the reference node
- The mode switches from observer to active or vice versa

Having CTSS running to synchronize time in a cluster facilitates troubleshooting Oracle Clusterware problems, because you will not have to factor in a time offset for a sequence of events on different nodes.

To activate CTSS in your cluster, you must stop and deconfigure the vendor time synchronization service on *all* nodes in the cluster. CTSS detects when this happens and assumes time management for the cluster.

For example, to deconfigure NTP, you must remove or rename the `etc/ntp.conf` file.

Similarly, to deactivate CTSS in your cluster:

1. Configure the vendor time synchronization service on *all* nodes in the cluster. CTSS detects this change and reverts back to observer mode.
2. Use the `crsctl check ctss` command to ensure that CTSS is operating in observer mode.
3. Start the vendor time synchronization service on *all* nodes in the cluster.
4. Use the `cluvfy comp clocksync -n all` command to verify that the vendor time synchronization service is operating.

> **See Also:** *Oracle Grid Infrastructure Installation Guide* for your platform for information about configuring NTP for Oracle Clusterware, or disabling it to use CTSS

2 Administering Oracle Clusterware

This chapter describes how to administer Oracle Clusterware. It includes the following topics:

- Role-Separated Management
- Overview of Grid Naming Service
- Configuring Oracle Grid Infrastructure Using Configuration Wizard
- Configuring IPMI for Failure Isolation
- Understanding Network Addresses on Manually Configured Networks
- Administering Grid Naming Service
- Rolling Conversion from DNS to GNS Cluster Name Resolution
- Changing Network Addresses on Manually Configured Systems

Role-Separated Management

This section includes the following topics

- About Role-Separated Management
- Managing Cluster Administrators in the Cluster
- Configuring Horizontal Role Separation

About Role-Separated Management

Role-separated management is a feature you can implement that enables multiple applications and databases to share the same cluster and hardware resources, in a coordinated manner, by setting permissions on server pools or resources, to provide or restrict access to resources, as required. By default, this feature is not implemented during installation.

You can implement role-separated management in one of two ways:

- **Vertical implementation** (between layers) describes a role separation approach based on different operating system users and groups used for various layers in the technology stack. Permissions on server pools and resources are granted to different users (and groups) for each layer in the stack using access control lists. Oracle Automatic Storage Management (ASM) offers setting up role separation as part of the Oracle Grid Infrastructure installation based on a granular assignment of operating system groups for specific roles.

- **Horizontal implementation** (within one layer) describes a role separation approach that restricts resource access within one layer using access permissions for resources that are granted using access control lists assigned to server pools and policy-managed databases or applications.

For example, consider an operating system user called `grid`, with primary operating system group `oinstall`, that installs Oracle Grid Infrastructure and creates two database server pools. The operating system users `ouser1` and `ouser2` must be able to operate within a server pool, but should not be able to modify those server pools so that hardware resources can be withdrawn from other server pools either accidentally or intentionally.

You can configure server pools before you deploy database software and databases by configuring a respective policy set.

> **See Also:** "Overview of Cluster Configuration Policies and the Policy Set" on page 3-9 for more information

Role-separated management in Oracle Clusterware no longer depends on a **cluster administrator** (but backward compatibility is maintained). By default, the user that installed Oracle Clusterware in the Oracle Grid Infrastructure home (Grid home) and `root` are *permanent* cluster administrators. Primary group privileges (`oinstall` by default) enable database administrators to create databases in newly created server pools using the Database Configuration Assistant (DBCA), but do not enable role separation.

> **Note:** Oracle recommends that you enable role separation before you create the first server pool in the cluster. Create and manage server pools using configuration policies and a respective policy set. Access permissions are stored for each server pool in the `ACL` attribute, described in Table 3–1, " Server Pool Attributes".

Managing Cluster Administrators in the Cluster

The ability to create server pools in a cluster is limited to the cluster administrators. In prior releases, by default, every registered operating system user was considered a cluster administrator and, if necessary, the default could be changed using `crsctl add | delete crs administrator` commands. The use of these commands, however, is deprecated in this release and, instead, you should use the access control list (ACL) of the policy set to control the ability to create server pools.

As a rule, to have permission to create a server pool, the operating system user or an operating system group of which the user is a member must have the read, write, and execute permissions set in the `ACL` attribute. Use the `crsctl modify policyset -attr "ACL=`*`value`*`"` command to add or remove permissions for operating system users and groups.

Configuring Horizontal Role Separation

Use the `crsctl setperm` command to configure horizontal role separation using ACLs that are assigned to server pools, resources, or both. The CRSCTL utility is located in the path *`Grid_home`*`/bin`, where *`Grid_home`* is the Oracle Grid Infrastructure for a cluster home.

The command uses the following syntax, where the access control (ACL) string is indicated by italics:

```
crsctl setperm {resource | type | serverpool} name {-u acl_string |
-x acl_string | -o user_name | -g group_name}
```

The flag options are:

- `-u`: Update the entity ACL
- `-x`: Delete the entity ACL
- `-o`: Change the entity owner
- `-g`: Change the entity primary group

The ACL strings are:

```
{ user:user_name[:readPermwritePermexecPerm]   |
     group:group_name[:readPermwritePermexecPerm] |
     other[::readPermwritePermexecPerm] }
```

where:

- `user`: Designates the user ACL (access permissions granted to the designated user)
- `group`: Designates the group ACL (permissions granted to the designated group members)
- `other`: Designates the other ACL (access granted to users or groups not granted particular access permissions)
- *`readperm`*: Location of the read permission (`r` grants permission and "`-`" forbids permission)
- *`writeperm`*: Location of the write permission (`w` grants permission and "`-`" forbids permission)
- *`execperm`*: Location of the execute permission (`x` grants permission, and "`-`" forbids permission)

For example, to set permissions on a server pool called `psft` for the group `personnel`, where the administrative user has read/write/execute privileges, the members of the `personnel` group have read/write privileges, and users outside of the group are granted no access, enter the following command as the `root` user:

```
# crsctl setperm serverpool psft -u user:personadmin:rwx,group:personnel:rw-,
  other::---
```

Overview of Grid Naming Service

Review the following sections to use Grid Naming Service (GNS) for address resolution:

- Network Administration Tasks for GNS and GNS Virtual IP Address
- Understanding Grid Naming Service Configuration Options

Network Administration Tasks for GNS and GNS Virtual IP Address

To implement GNS, your network administrator must configure the DNS to set up a domain for the cluster, and delegate resolution of that domain to the GNS VIP. You can use a separate domain, or you can create a subdomain of an existing domain for the cluster.

GNS distinguishes between nodes by using cluster names and individual node identifiers as part of the host name for that cluster node, so that cluster node 123 in cluster A is distinguishable from cluster node 123 in cluster B.

However, if you configure host names manually, then the subdomain you delegate to GNS should have no subdomains. For example, if you delegate the subdomain mydomain.example.com to GNS for resolution, then there should be no other.mydomain.example.com domains. Oracle recommends that you delegate a subdomain to GNS that is used by GNS exclusively.

Note: You can use GNS without DNS delegation in configurations where static addressing is being done, such as in Oracle Flex ASM or Oracle Flex Clusters. However, GNS requires a domain be delegated to it if addresses are assigned using DHCP.

Example 2–1 shows DNS entries required to delegate a domain called myclustergns.example.com to a GNS VIP address 10.9.8.7:

Example 2–1 DNS Entries

```
# Delegate to gns on mycluster
mycluster.example.com NS myclustergns.example.com
#Let the world know to go to the GNS vip
myclustergns.example.com. 10.9.8.7
```

See Also: *Oracle Grid Infrastructure Installation Guide* for more information about network domains and delegation for GNS

The GNS daemon and the GNS VIP run on one node in the server cluster. The GNS daemon listens on the GNS VIP using port 53 for DNS requests. Oracle Clusterware manages the GNS daemon and the GNS VIP to ensure that they are always available. If the server on which the GNS daemon is running fails, then Oracle Clusterware fails over the GNS daemon and the GNS VIP to a surviving cluster member node. If the cluster is an Oracle Flex Cluster configuration, then Oracle Clusterware fails over the GNS daemon and the GNS VIP to a Hub Node.

Note: Oracle Clusterware does not fail over GNS addresses to different clusters. Failovers occur only to members of the same cluster.

See Also: Chapter 4, "Oracle Flex Clusters" for more information about Oracle Flex Clusters and GNS

Understanding Grid Naming Service Configuration Options

GNS can run in either *automatic* or *standard* cluster address configuration mode. Automatic configuration uses either the Dynamic Host Configuration Protocol (DHCP) for IPv4 addresses or the Stateless Address Autoconfiguration Protocol (autoconfig) (RFC 2462 and RFC 4862) for IPv6 addresses.

This section includes the following topics:

- Automatic Configuration Option for Addresses
- Static Configuration Option for Addresses
- Shared GNS Option for Addresses

Automatic Configuration Option for Addresses

With automatic configurations, a DNS administrator delegates a domain on the DNS to be resolved through the GNS subdomain. During installation, Oracle Universal Installer assigns names for each cluster member node interface designated for Oracle Grid Infrastructure use during installation or configuration. SCANs and all other cluster names and addresses are resolved within the cluster, rather than on the DNS.

Automatic configuration occurs in one of the following ways:

- For IPv4 addresses, Oracle Clusterware assigns unique identifiers for each cluster member node interface allocated for Oracle Grid Infrastructure, and generates names using these identifiers within the subdomain delegated to GNS. A DHCP server assigns addresses to these interfaces, and GNS maintains address and name associations with the IPv4 addresses leased from the IPv4 DHCP pool.
- For IPv6 addresses, Oracle Clusterware automatically generates addresses with autoconfig.

Static Configuration Option for Addresses

With static configurations, no subdomain is delegated. A DNS administrator configures the GNS VIP to resolve to a name and address configured on the DNS, and a DNS administrator configures a SCAN name to resolve to three static addresses for the cluster. A DNS administrator also configures a static public IP name and address, and virtual IP name and address for each cluster member node. A DNS administrator must also configure new public and virtual IP names and addresses for each node added to the cluster. All names and addresses are resolved by DNS.

GNS without subdomain delegation using static VIP addresses and SCANs enables Oracle Flex Cluster and CloudFS features that require name resolution information within the cluster. However, any node additions or changes must be carried out as manual administration tasks.

Shared GNS Option for Addresses

With dynamic configurations, you can configure GNS to provide name resolution for one cluster, or to advertise resolution for multiple clusters, so that a single GNS instance can perform name resolution for multiple registered clusters. This option is called shared GNS.

Note: All of the node names in a set of clusters served by GNS must be unique.

Shared GNS provides the same services as standard GNS, and appears the same to clients receiving name resolution. The difference is that the GNS daemon running on one cluster is configured to provide name resolution for all clusters in domains that are delegated to GNS for resolution, and GNS can be centrally managed using SRVCTL commands. You can use shared GNS configuration to minimize network administration tasks across the enterprise for Oracle Grid Infrastructure clusters.

You cannot use the static address configuration option for a cluster providing shared GNS to resolve addresses in a multi-cluster environment. Shared GNS requires automatic address configuration, either through addresses assigned by DHCP, or by IPv6 stateless address autoconfiguration.

Oracle Universal Installer enables you to configure static addresses with GNS for shared GNS clients or servers, with GNS used for discovery.

Configuring Oracle Grid Infrastructure Using Configuration Wizard

After performing a software-only installation of the Oracle Grid Infrastructure, you can configure the software using Configuration Wizard. This wizard assists you with editing the `crsconfig_params` configuration file. Similar to the Oracle Grid Infrastructure installer, the Configuration Wizard performs various validations of the Grid home and inputs before and after you run through the wizard.

Using the Configuration Wizard, you can configure a new Oracle Grid Infrastructure on one or more nodes, or configure an upgraded Oracle Grid Infrastructure. You can also run the Configuration Wizard in silent mode.

Notes:

- Before running the Configuration Wizard, ensure that the Oracle Grid Infrastructure home is current, with all necessary patches applied.
- To launch the Configuration Wizard in the following procedures:

  On Linux and UNIX, run the following command:

  ```
  Oracle_home/crs/config/config.sh
  ```

  On Windows, run the following command:

  ```
  Oracle_home\crs\config\config.bat
  ```

This section includes the following topics:

- Configuring a Single Node
- Configuring Multiple Nodes
- Upgrading Oracle Grid Infrastructure
- Running the Configuration Wizard in Silent Mode

Configuring a Single Node

To use the Configuration Wizard to configure a single node:

1. Start the Configuration Wizard, as follows:

   ```
   $ Oracle_home/crs/config/config.sh
   ```

2. On the Select Installation Option page, select **Configure Oracle Grid Infrastructure for a Cluster**.
3. On the Cluster Node Information page, select only the local node and corresponding VIP name.
4. Continue adding your information on the remaining wizard pages.
5. Review your inputs on the Summary page and click **Finish**.
6. Run the `root.sh` script as instructed by the Configuration Wizard.

Configuring Multiple Nodes

Note: Before you launch the Configuration Wizard, ensure the following:

- The Oracle Grid Infrastructure software is installed on all the target nodes
- The software is installed in the same *`Grid_home`* path on all the nodes
- The software level (including patches) is identical on all the nodes

To use the Configuration Wizard to configure multiple nodes:

1. Start the Configuration Wizard, as follows:

   ```
   $ Oracle_home/crs/config/config.sh
   ```

2. On the Select Installation Option page, select **Configure Oracle Grid Infrastructure for a Cluster**.
3. On the Cluster Node Information page, select the nodes you want to configure and their corresponding VIP names. The Configuration Wizard validates the nodes you select to ensure that they are ready.
4. Continue adding your information on the remaining wizard pages.
5. Review your inputs on the Summary page and click **Finish**.
6. Run the `root.sh` script as instructed by the Configuration Wizard.

Upgrading Oracle Grid Infrastructure

To use the Configuration Wizard to upgrade Oracle Grid Infrastructure for a cluster:

1. Start the Configuration Wizard:

   ```
   $ Oracle_home/crs/config/config.sh
   ```

2. On the Select Installation Option page, select **Upgrade Oracle Grid Infrastructure**.
3. On the Oracle Grid Infrastructure Node Selection page, select the nodes you want to upgrade.
4. Continue adding your information on the remaining wizard pages.
5. Review your inputs on the Summary page and click **Finish**.
6. Run the `rootupgrade.sh` script as instructed by the Configuration Wizard.

Note: Oracle Restart cannot be upgraded using the Configuration Wizard.

See Also: *Oracle Database Installation Guide* for your platform for Oracle Restart procedures

Running the Configuration Wizard in Silent Mode

To use the Configuration Wizard in silent mode to configure or upgrade nodes, start the Configuration Wizard from the command line with `-silent -responseFile` *`file_name`*. The wizard validates the response file and proceeds with the configuration. If any of the inputs in the response file are found to be invalid, then the Configuration Wizard displays an error and exits. Run the `root` and `configToolAllCommands` scripts as prompted.

Configuring IPMI for Failure Isolation

This section contains the following topics:

- About Using IPMI for Failure Isolation
- Configuring Server Hardware for IPMI
- Postinstallation Configuration of IPMI-based Failure Isolation Using CRSCTL

About Using IPMI for Failure Isolation

Failure isolation is a process by which a failed node is isolated from the rest of the cluster to prevent the failed node from corrupting data. The ideal fencing involves an external mechanism capable of restarting a problem node without cooperation either from Oracle Clusterware or from the operating system running on that node. To provide this capability, Oracle Clusterware 12*c* supports the Intelligent Platform Management Interface specification (IPMI) (also known as Baseboard Management Controller (BMC)), an industry-standard management protocol.

Typically, you configure failure isolation using IPMI during Oracle Grid Infrastructure installation, when you are provided with the option of configuring IPMI from the Failure Isolation Support screen. If you do not configure IPMI during installation, then you can configure it after installation using the Oracle Clusterware Control utility (CRSCTL), as described in "Postinstallation Configuration of IPMI-based Failure Isolation Using CRSCTL" on page 2-9.

To use IPMI for failure isolation, each cluster member node must be equipped with an IPMI device running firmware compatible with IPMI version 1.5, which supports IPMI over a local area network (LAN). During database operation, failure isolation is accomplished by communication from the evicting Cluster Synchronization Services daemon to the failed node's IPMI device over the LAN. The IPMI-over-LAN protocol is carried over an authenticated session protected by a user name and password, which are obtained from the administrator during installation.

To support dynamic IP address assignment for IPMI using DHCP, the Cluster Synchronization Services daemon requires direct communication with the local IPMI device during Cluster Synchronization Services startup to obtain the IP address of the IPMI device. (This is not true for HP-UX and Solaris platforms, however, which require that the IPMI device be assigned a static IP address.) This is accomplished using an IPMI probe command (OSD), which communicates with the IPMI device through an IPMI driver, which you must install on each cluster system.

If you assign a static IP address to the IPMI device, then the IPMI driver is not strictly required by the Cluster Synchronization Services daemon. The driver *is* required, however, to use `ipmitool` or `ipmiutil` to configure the IPMI device but you can also do this with management consoles on some platforms.

Configuring Server Hardware for IPMI

Install and enable the IPMI driver, and configure the IPMI device, as described in the *Oracle Grid Infrastructure Installation Guide* for your platform.

Postinstallation Configuration of IPMI-based Failure Isolation Using CRSCTL

This section contains the following topics:

- IPMI Postinstallation Configuration with Oracle Clusterware
- Modifying IPMI Configuration Using CRSCTL
- Removing IPMI Configuration Using CRSCTL

IPMI Postinstallation Configuration with Oracle Clusterware

When you install IPMI during Oracle Clusterware installation, you configure failure isolation in two phases. Before you start the installation, you install and enable the IPMI driver in the server operating system, and configure the IPMI hardware on each node (IP address mode, admin credentials, and so on), as described in *Oracle Grid Infrastructure Installation Guide*. When you install Oracle Clusterware, the installer collects the IPMI administrator user ID and password, and stores them in an Oracle Wallet in node-local storage, in OLR.

After you complete the server configuration, complete the following procedure on each cluster node to register IPMI administrators and passwords on the nodes.

> **Note:** If IPMI is configured to obtain its IP address using DHCP, it may be necessary to reset IPMI or restart the node to cause it to obtain an address.

1. Start Oracle Clusterware, which allows it to obtain the current IP address from IPMI. This confirms the ability of the clusterware to communicate with IPMI, which is necessary at startup.

   If Oracle Clusterware was running before IPMI was configured, you can shut Oracle Clusterware down and restart it. Alternatively, you can use the IPMI management utility to obtain the IPMI IP address and then use CRSCTL to store the IP address in OLR by running a command similar to the following:

   ```
   crsctl set css ipmiaddr 192.168.10.45
   ```

2. Use CRSCTL to store the previously established user ID and password for the resident IPMI in OLR by running the `crsctl set css ipmiadmin` command, and supplying password at the prompt. For example:

   ```
   crsctl set css ipmiadmin administrator_name
   IPMI BMC password: password
   ```

   This command validates the supplied credentials and fails if another cluster node cannot access the local IPMI using them.

   After you complete hardware and operating system configuration, and register the IPMI administrator on Oracle Clusterware, IPMI-based failure isolation should be fully functional.

Modifying IPMI Configuration Using CRSCTL

To modify an existing IPMI-based failure isolation configuration (for example to change IPMI passwords, or to configure IPMI for failure isolation in an existing installation), use CRSCTL with the IPMI configuration tool appropriate to your platform. For example, to change the administrator password for IPMI, you must first modify the IMPI configuration as described in *Oracle Grid Infrastructure Installation Guide*, and then use CRSCTL to change the password in OLR.

The configuration data needed by Oracle Clusterware for IPMI is kept in an Oracle Wallet in OCR. Because the configuration information is kept in a secure store, it must be written by the Oracle Clusterware installation owner account (the Grid user), so you must log in as that installation user.

Use the following procedure to modify an existing IPMI configuration:

1. Enter the `crsctl set css ipmiadmin` *`administrator_name`* command. For example, with the user `IPMIadm`:

   ```
   crsctl set css ipmiadmin IPMIadm
   ```

   Provide the administrator password. Oracle Clusterware stores the administrator name and password for the local IPMI in OLR.

   After storing the new credentials, Oracle Clusterware can retrieve the new credentials and distribute them as required.

2. Enter the `crsctl set css ipmiaddr` *`bmc_ip_address`* command. For example:

   ```
   crsctl set css ipmiaddr 192.0.2.244
   ```

   This command stores the new IPMI IP address of the local IPMI in OLR, After storing the IP address, Oracle Clusterware can retrieve the new configuration and distribute it as required.

3. Enter the `crsctl get css ipmiaddr` command. For example:

   ```
   crsctl get css ipmiaddr
   ```

   This command retrieves the IP address for the local IPMI from OLR and displays it on the console.

4. Remove the IPMI configuration information for the local IPMI from OLR and delete the registry entry, as follows:

   ```
   crsctl unset css ipmiconfig
   ```

> **See Also:** "Oracle RAC Environment CRSCTL Commands" on page E-36 for descriptions of these CRSCTL commands

Removing IPMI Configuration Using CRSCTL

You can remove an IPMI configuration from a cluster using CRSCTL if you want to stop using IPMI completely or if IPMI was initially configured by someone other than the user that installed Oracle Clusterware. If the latter is true, then Oracle Clusterware cannot access the IPMI configuration data and IPMI is not usable by the Oracle Clusterware software, and you must reconfigure IPMI as the user that installed Oracle Clusterware.

To completely remove IPMI, perform the following steps. To reconfigure IPMI as the user that installed Oracle Clusterware, perform steps 3 and 4, then repeat steps 2 and 3 in "Modifying IPMI Configuration Using CRSCTL" on page 2-10.

1. Disable the IPMI driver and eliminate the boot-time installation, as follows:

```
/sbin/modprobe -r
```

> **See Also:** *Oracle Grid Infrastructure Installation Guide* for your platform for more information about the IPMI driver

2. Disable IPMI-over-LAN for the local IPMI using either `ipmitool` or `ipmiutil`, to prevent access over the LAN or change the IPMI administrator user ID and password.
3. Ensure that Oracle Clusterware is running and then use CRSCTL to remove the IPMI configuration data from OLR by running the following command:

   ```
   crsctl unset css ipmiconfig
   ```

4. Restart Oracle Clusterware so that it runs without the IPMI configuration by running the following commands as `root`:

   ```
   # crsctl stop crs
   # crsctl start crs
   ```

Understanding Network Addresses on Manually Configured Networks

This section contains the following topics:

- Understanding Network Address Configuration Requirements
- Understanding SCAN Addresses and Client Service Connections
- SCAN Listeners and Service Registration Restriction With Valid Node Checking

Understanding Network Address Configuration Requirements

An Oracle Clusterware configuration requires at least two interfaces:

- A public network interface, on which users and application servers connect to access data on the database server
- A private network interface for internode communication.

You can configure a network interface for either **IPv4**, **IPv6**, or both types of addresses on a given network. If you use redundant network interfaces (bonded or teamed interfaces), then be aware that Oracle does not support configuring one interface to support IPv4 addresses and the other to support IPv6 addresses. You must configure network interfaces of a redundant interface pair with the same IP protocol.

All the nodes in the cluster must use the same IP protocol configuration. Either all the nodes use only IPv4, or all the nodes use only IPv6, or all the nodes use both IPv4 and IPv6. You cannot have some nodes in the cluster configured to support only IPv6 addresses, and other nodes in the cluster configured to support only IPv4 addresses.

The VIP agent supports the generation of IPv6 addresses using the Stateless Address Autoconfiguration Protocol (RFC 2462), and advertises these addresses with GNS. Run the `srvctl config network` command to determine if DHCP or stateless address autoconfiguration is being used.

This section includes the following topics:

- About IPv6 Address Formats
- Name Resolution and the Network Resource Address Type

About IPv6 Address Formats

Each node in an Oracle Grid Infrastructure cluster can support both IPv4 and IPv6 addresses on the same network. The preferred IPv6 address format is as follows, where each x represents a hexadecimal character:

```
xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
```

The IPv6 address format is defined by RFC 2460 and Oracle Grid Infrastructure supports IPv6 addresses, as following:

- Global and site-local IPv6 addresses as defined by RFC 4193.

> **Note:** Link-local and site-local IPv6 addresses as defined in RFC 1884 *are not* supported.

- The leading zeros compressed in each field of the IP address.
- Empty fields collapsed and represented by a '::' separator. For example, you could write the IPv6 address 2001:0db8:0000:0000:0000:8a2e:0370:7334 as 2001:db8::8a2e:370:7334.
- The four lower order fields containing 8-bit pieces (standard IPv4 address format). For example 2001:db8:122:344::192.0.2.33.

Name Resolution and the Network Resource Address Type

You can review the network configuration and control the network address type using the `srvctl config network` (to review the configuration) or `srvctl status network` (to review the current addresses allocated for dynamic networks), and `srvctl modify network -iptype` commands, respectively.

You can configure how addresses are acquired using the `srvctl modify network -nettype` command. Set the value of the `-nettype` parameter to `dhcp` or `static` to control how IPv4 network addresses are acquired. Alternatively, set the value of the `-nettype` parameter to `autoconfig` or `static` to control how IPv6 addresses are generated.

The `-nettype` and `-iptype` parameters are not directly related but you can use `-nettype dhcp` with `-iptype ipv4` and `-nettype autoconfig` with `-iptype ipv6`.

> **Note:** If a network is configured with both IPv4 and IPv6 subnets, then Oracle does not support both subnets having `-nettype` set to `mixed`.
>
> Oracle does not support making transitions from IPv4 to IPv6 while `-nettype` is set to `mixed`. You must first finish the transition from `static` to `dhcp` before you add IPv6 into the subnet.
>
> Similarly, Oracle does not support starting a transition to IPv4 from IPv6 while `-nettype` is set to `mixed`. You must first finish the transition from `autoconfig` to `static` before you add IPv4 into the subnet.

See Also:

- *Oracle Real Application Clusters Administration and Deployment Guide* for more information about the `srvctl modify network` command
- "Changing Network Address Types Using SRVCTL" on page 2-26 for information about using SRVCTL to report and manage network resource attribute settings

Understanding SCAN Addresses and Client Service Connections

Public network addresses are used to provide services to clients. If your clients are connecting to the Single Client Access Name (SCAN) addresses, then you may need to change public and virtual IP addresses as you add or remove nodes from the cluster, but you do not need to update clients with new cluster addresses.

Note: You can edit the `listener.ora` file to make modifications to the Oracle Net listener parameters for SCAN and the node listener. For example, you can set `TRACE_LEVEL_`*`listener_name`*. However, you cannot set protocol address parameters to define listening endpoints, because the listener agent dynamically manages them.

See Also: *Oracle Database Net Services Reference* for more information about the editing the `listener.ora` file

SCANs function like a cluster alias. However, SCANs are resolved on any node in the cluster, so unlike a VIP address for a node, clients connecting to the SCAN no longer require updated VIP addresses as nodes are added to or removed from the cluster. Because the SCAN addresses resolve to the cluster, rather than to a node address in the cluster, nodes can be added to or removed from the cluster without affecting the SCAN address configuration.

The SCAN is a fully qualified name (host name and domain) that is configured to resolve to all the addresses allocated for the SCAN. The SCAN resolves to one of the three addresses configured for the SCAN name on the DNS server, or resolves within the cluster in a GNS configuration. SCAN listeners can run on any node in the cluster. SCANs provide location independence for the databases, so that client configuration does not have to depend on which nodes run a particular database.

Oracle Database 11*g* release 2 (11.2), and later, instances only register with SCAN listeners as remote listeners. Upgraded databases register with SCAN listeners as remote listeners, and also continue to register with all node listeners.

Note: Because of the Oracle Clusterware installation requirement that you provide a SCAN name during installation, if you resolved at least one IP address using the server `/etc/hosts` file to bypass the installation requirement but you do not have the infrastructure required for SCAN, then, after the installation, you can ignore the SCAN and connect to the databases in the cluster using VIPs.

Oracle *does not* support removing the SCAN address.

SCAN Listeners and Service Registration Restriction With Valid Node Checking

You can use valid node checking to specify the nodes and subnets from which the SCAN listener accepts registrations. You can specify the nodes and subnet information using SRVCTL. SRVCTL stores the node and subnet information in the SCAN listener resource profile. The SCAN listener agent reads that information from the resource profile and writes it to the `listener.ora` file.

For non-cluster (single-instance) databases, the local listener accepts service registrations only from database instances on the local node. Oracle RAC releases before Oracle RAC 11*g* release 2 (11.2) do not use SCAN listeners, and attempt to register their services with the local listener and the listeners defined by the `REMOTE_LISTENERS` initialization parameter. To support service registration for these database instances, the default value of `valid_node_check_for_registration_`*`alias`* for the local listener in Oracle RAC 12*c* is set to the value `SUBNET`, rather than to the local node. To change the valid node checking settings for the node listeners, edit the `listener.ora` file.

SCAN listeners must accept service registration from instances on remote nodes. For SCAN listeners, the value of `valid_node_check_for_registration_`*`alias`* is set to `SUBNET` in the `listener.ora` file so that the corresponding listener can accept service registrations that originate from the same subnet.

You can configure the listeners to accept service registrations from a different subnet. For example, you might want to configure this environment when SCAN listeners share with instances on different clusters, and nodes in those clusters are on a different subnet. Run the `srvctl modfiy scan_listener -invitednodes -invitedsubnets` command to include the nodes in this environment.

You must also run the `srvctl modify nodeapps -remoteservers` *`host`*`:`*`port`*`,...` command to connect the Oracle Notification Service networks of this cluster and the cluster with the invited instances.

See Also:

- The `srvctl add scan_listener`, `srvctl modify nodeapps`, and `svrctl modify scan_listener` commands in *Oracle Real Application Clusters Administration and Deployment Guide* for detailed usage information
- *Oracle Database Net Services Administrator's Guide*

Administering Grid Naming Service

Use SRVCTL to administer Grid Naming Service (GNS) in both single-cluster and multi-cluster environments.

This section includes the following topics:

- Starting and Stopping GNS with SRVCTL
- Converting Clusters to GNS Server or GNS Client Clusters
- Moving GNS to Another Cluster

Note: The GNS server and client must run on computers using the same operating system and processor architecture. Oracle does not support running GNS on computers with different operating systems, processor architectures, or both.

See Also: *Oracle Real Application Clusters Administration and Deployment Guide* for usage information for the SRVCTL commands used in the procedures described in this section

Starting and Stopping GNS with SRVCTL

Start and stop GNS on the server cluster by running the following commands as `root`, respectively:

```
# srvctl start gns
# srvctl stop gns
```

Note: You cannot start or stop GNS on a client cluster.

Converting Clusters to GNS Server or GNS Client Clusters

You can convert clusters that are not running GNS into GNS server or client clusters, and you can change GNS cluster type configurations for server and client clusters.

This section includes the following cluster conversion scenarios:

- Converting a Non-GNS Cluster to a GNS Server Cluster
- Converting a Non-GNS Cluster to a Client Cluster
- Converting a Single Cluster Running GNS to a Server Cluster
- Converting a Single Cluster Running GNS to be a GNS Client Cluster

Converting a Non-GNS Cluster to a GNS Server Cluster

To convert a cluster that is not running GNS to a GNS server cluster, run the following command as `root`, providing a valid IP address and a domain:

```
# srvctl add gns -vip IP_address -domain domain
```

Notes:

- Specifying a domain is *not* required when adding a GNS VIP.
- The IP address you specify cannot currently be used by another GNS instance.
- The configured cluster must have DNS delegation for it to be a GNS server cluster.

Converting a Non-GNS Cluster to a Client Cluster

To convert a cluster that is not running GNS to a GNS client cluster:

1. Log in as `root` and run the following command in the server cluster to export the GNS instance client data configuration to a file:

   ```
   # srvctl export gns -clientdata path_to_file
   ```

   You must specify the fully-qualified path to the file.

Note: You can use the GNS configuration Client Data file you generate with Oracle Universal Installer as an input file for creating shared GNS clients.

2. Import the file you created in the preceding step on a node in the cluster to make that cluster a client cluster by running the following command as `root`:

   ```
   # srvctl add gns -clientdata path_to_file
   ```

   > **Note:** You must copy the file containing the GNS data from the server cluster to a node in the cluster where you run this command.

3. Change the SCAN name, as follows:

   ```
   $ srvctl modify scan -scanname scan.client_clustername.server_GNS_subdomain
   ```

Converting a Single Cluster Running GNS to a Server Cluster

You do not need to do anything to convert a single cluster running GNS to be a GNS server cluster. It is automatically considered to be a server cluster when a client cluster is added.

Converting a Single Cluster Running GNS to be a GNS Client Cluster

Because it is necessary to stay connected to the current GNS during this conversion process, the procedure is more involved than that of converting a single cluster to a server cluster.

1. Run the following command as `root` in the server cluster to export the GNS client information to a file:

   ```
   # srvctl export gns -clientdata path_to_client_data_file
   ```

   You must specify the fully-qualified path to the file.

2. Stop GNS on the cluster you want to convert to a client cluster.

   ```
   # srvctl stop gns
   ```

   > **Note:** While the conversion is in progress, name resolution using GNS will be unavailable.

3. Run the following command as `root` in the server cluster to export the GNS instance:

   ```
   # srvctl export gns -instance path_to_file
   ```

   You must specify the fully-qualified path to the file.

4. Run the following command as `root` in the server cluster to import the GNS instance file:

   ```
   # srvctl import gns -instance path_to_file
   ```

   You must specify the fully-qualified path to the file.

5. Run the following command as `root` on the node where you imported the GNS instance file to start the GNS instance:

   ```
   # srvctl start gns
   ```

   By not specifying the name of the node on which you want to start the GNS instance, the instance will start on a random node.

6. Remove GNS from the GNS client cluster using the following command:

```
# srvctl remove gns
```

7. Make the former cluster a client cluster, as follows:

   ```
   # srvctl add gns -clientdata path_to_client_data_file
   ```

   > **Note:** You must copy the file containing the GNS data from the server cluster to a node in the cluster where you run this command.

8. Modify the SCAN in the GNS client cluster to use the GNS subdomain qualified with the client cluster name, as follows:

   ```
   srvctl modify scan -scanname scan_name.gns_domain
   ```

   In the preceding command, *`gns_domain`* is in the form *`client clustername.server GNS subdomain`*

Moving GNS to Another Cluster

> **Note:** This procedure requires server cluster and client cluster downtime. Additionally, you must import GNS client data from the new server cluster to any Oracle Flex ASM and Grid Home servers and clients.

If it becomes necessary to make another cluster the GNS server cluster, either because a cluster failure, or because of an administration plan, then you can move GNS to another cluster using the following procedure:

1. Stop the GNS instance on the current server cluster.

   ```
   # srvctl stop gns
   ```

2. Export the GNS instance configuration to a file.

   ```
   # srvctl export gns -instance path_to_file
   ```

   Specify the fully-qualified path to the file.

3. Remove the GNS configuration from the former server cluster.

   ```
   # srvctl remove gns
   ```

4. Add GNS to the new cluster.

   ```
   # srvctl add gns -domain domain_name -vip vip_name
   ```

   Alternatively, you can specify an IP address for the VIP.

5. Configure the GNS instance in the new server cluster using the instance information stored in the file you created in step 2, by importing the file, as follows:

   ```
   # srvctl import gns -instance path_to_file
   ```

   > **Note:** The file containing the GNS data from the former server cluster must reside on the node in the cluster where you run the `srvctl import gns` command.

6. Start the GNS instance in the new server cluster.

   ```
   # srvctl start gns
   ```

Rolling Conversion from DNS to GNS Cluster Name Resolution

You can convert Oracle Grid Infrastructure cluster networks using DNS for name resolution to cluster networks using Grid Naming Service (GNS) obtaining name resolution through GNS.

Use the following procedure to convert from a standard DNS name resolution network to a GNS name resolution network, with no downtime:

> **See Also:** *Oracle Grid Infrastructure Installation Guide* for your platform to complete preinstallation steps for configuring GNS

1. Log in as the Grid user (`grid`), and use the following Cluster Verification Utility to check the status for moving the cluster to GNS, where *`nodelist`* is a comma-delimited list of cluster member nodes:

   ```
   $ cluvfy stage -pre crsinst -n nodelist
   ```

2. As the Grid user, check the integrity of the GNS configuration using the following commands, where *`domain`* is the domain delegated to GNS for resolution, and *`gns_vip`* is the GNS VIP:

   ```
   $ cluvfy comp gns -precrsinst -domain domain -vip gns_vip
   ```

3. Log in as `root`, and use the following SRVCTL command to configure the GNS resource, where *`domain_name`* is the domain that your network administrator has configured your DNS to delegate for resolution to GNS, and *`ip_address`* is the IP address on which GNS listens for DNS requests:

   ```
   # srvctl add gns -domain domain_name -vip ip_address
   ```

4. Use the following command to start GNS:

   ```
   # srvctl start gns
   ```

   GNS starts and registers VIP and SCAN names.

5. As `root`, use the following command to change the network CRS resource to support a mixed mode of static and DHCP network addresses:

   ```
   # srvctl modify network -nettype MIXED
   ```

   The necessary VIP addresses are obtained from the DHCP server, and brought up.

6. As the Grid user, enter the following command to ensure that Oracle Clusterware is using the new GNS, dynamic addresses, and listener end points:

   ```
   cluvfy stage -post crsinst -n all
   ```

7. After the verification succeeds, change the remote endpoints that previously used the SCAN or VIPs resolved through the DNS to use the SCAN and VIPs resolved through GNS.

   For each client using a SCAN, change the SCAN that the client uses so that the client uses the SCAN in the domain delegated to GNS.

For each client using VIP names, change the VIP name on each client so that they use the same server VIP name, but with the domain name in the domain delegated to GNS.

8. Enter the following command as `root` to update the system with the SCAN name in the GNS subdomain:

   ```
   # srvctl modify scan -scanname scan_name.gns_domain
   ```

   In the preceding command syntax, *gns_domain* is the domain name you entered in step 3 of this procedure.

9. Disable the static addresses once all clients are using the dynamic addresses, as follows:

   ```
   $ srvctl modify network -nettype DHCP
   ```

Changing Network Addresses on Manually Configured Systems

This section includes the following topics:

- Changing the Virtual IP Addresses Using SRVCTL
- Changing Oracle Clusterware Private Network Configuration
- Creating a Network Using SRVCTL
- Changing Network Address Types Using SRVCTL
- Changing Static IPv4 Addresses To Static IPv6 Addresses Using SRVCTL
- Changing Dynamic IPv4 Addresses To Dynamic IPv6 Addresses Using SRVCTL
- Changing an IPv4 Network to an IPv4 and IPv6 Network
- Transitioning from IPv4 to IPv6 Networks for VIP Addresses Using SRVCTL

Changing the Virtual IP Addresses Using SRVCTL

Clients configured to use public VIP addresses for Oracle Database releases before Oracle Database 11*g* release 2 (11.2) can continue to use their existing connection addresses. Oracle recommends that you configure clients to use SCANs, but you are not required to use SCANs. When an earlier version of Oracle Database is upgraded, it is registered with the SCAN, and clients can start using the SCAN to connect to that database, or continue to use VIP addresses for connections.

If you continue to use VIP addresses for client connections, you can modify the VIP address while Oracle Database and Oracle ASM continue to run. However, you must stop services while you modify the address. When you restart the VIP address, services are also restarted on the node.

You cannot use this procedure to change a static public subnet to use DHCP. Only the `srvctl add network -subnet` command creates a DHCP network.

> **See Also:** *Oracle Real Application Clusters Administration and Deployment Guide* for more information about the `srvctl add network` command

---

Note: The following instructions describe how to change only a VIP address, and assume that the host name associated with the VIP address does not change. Note that you do not need to update VIP addresses manually if you are using GNS, and VIPs are assigned using DHCP.

If you are changing only the VIP address, then update the DNS and the client hosts files. Also, update the server hosts files, if those are used for VIP addresses.

Perform the following steps to change a VIP address:

1. Stop all services running on the node whose VIP address you want to change using the following command syntax, where *database_name* is the name of the database, *service_name_list* is a list of the services you want to stop, and *my_node* is the name of the node whose VIP address you want to change:

   ```
   srvctl stop service -db database_name -service "service_name_list" -node node_
   name
   ```

   The following example specifies the database name (`grid`) using the `-db` option and specifies the services (`sales,oltp`) on the appropriate node (`mynode`).

   ```
   $ srvctl stop service -db grid -service "sales,oltp" -node mynode
   ```

2. Confirm the current IP address for the VIP address by running the `srvctl config vip` command. This command displays the current VIP address bound to one of the network interfaces. The following example displays the configured VIP address for a VIP named `node03-vip`:

   ```
   $ srvctl config vip -vipname node03-vip
   VIP exists: /node03-vip/192.168.2.20/255.255.255.0/eth0
   ```

3. Stop the VIP resource using the `srvctl stop vip` command:

   ```
   $ srvctl stop vip -node node_name
   ```

4. Verify that the VIP resource is no longer running by running the `ifconfig -a` command on Linux and UNIX systems (or issue the `ipconfig /all` command on Windows systems), and confirm that the interface (in the example it was `eth0:1`) is no longer listed in the output.

5. Make any changes necessary to the `/etc/hosts` files on all nodes on Linux and UNIX systems, or the `%windir%\system32\drivers\etc\hosts` file on Windows systems, and make any necessary DNS changes to associate the new IP address with the old host name.

6. To use a different subnet or network interface card for the default network before you change any VIP resource, you must use the `srvctl modify network -subnet` *`subnet`*`/`*`netmask`*`/`*`interface`* command as `root` to change the network resource, where *`subnet`* is the new subnet address, *`netmask`* is the new netmask, and *`interface`* is the new interface. After you change the subnet, then you must change each node's VIP to an IP address on the new subnet, as described in the next step.

7. Modify the node applications and provide the new VIP address using the following `srvctl modify nodeapps` syntax:

   ```
   $ srvctl modify nodeapps -node node_name -address new_vip_address
   ```

The command includes the following flags and values:

- `-n` *`node_name`* is the node name
- `-A` *`new_vip_address`* is the node-level VIP address: *`name`*`|`*`ip`*`/`*`netmask`*`/[`*`if1`*`[|`*`if2`*`|...]]`

  For example, issue the following command as the `root` user:

  ```
  srvctl modify nodeapps -node mynode -address
  192.168.2.125/255.255.255.0/eth0
  ```

  Attempting to issue this command as the installation owner account may result in an error. For example, if the installation owner is `oracle`, then you may see the error `PRCN-2018: Current user oracle is not a privileged user`.

  To avoid the error, run the command as the `root` or system administrator account.

8. Start the node VIP by running the `srvctl start vip` command:

   ```
   $ srvctl start vip -node node_name
   ```

   The following command example starts the VIP on the node named `mynode`:

   ```
   $ srvctl start vip -node mynode
   ```

9. Repeat the steps for each node in the cluster.

   Because the SRVCTL utility is a clusterwide management tool, you can accomplish these tasks for any specific node from any node in the cluster, without logging in to each of the cluster nodes.

10. Run the following command to verify node connectivity between all of the nodes for which your cluster is configured. This command discovers all of the network interfaces available on the cluster nodes and verifies the connectivity between all of the nodes by way of the discovered interfaces. This command also lists all of the interfaces available on the nodes which are suitable for use as VIP addresses.

    ```
    $ cluvfy comp nodecon -n all -verbose
    ```

Changing Oracle Clusterware Private Network Configuration

This section contains the following topics:

- About Private Networks and Network Interfaces
- Redundant Interconnect Usage
- Consequences of Changing Interface Names Using OIFCFG
- Changing a Network Interface

About Private Networks and Network Interfaces

Oracle Clusterware requires that each node is connected through a private network (in addition to the public network). The private network connection is referred to as the *cluster interconnect*. Table 2–1 describes how the network interface card and the private IP address are stored.

Oracle only supports clusters in which all of the nodes use the same network interface connected to the same subnet (defined as a global interface with the `oifcfg` command). You cannot use different network interfaces for each node (node-specific interfaces). Refer to Appendix D, "Oracle Interface Configuration Tool (OIFCFG) Command Reference" for more information about global and node-specific interfaces.

Table 2–1 Storage for the Network Interface, Private IP Address, and Private Host Name

Entity	Stored In...	Comments
Network interface name	Operating system For example: `eth1`	You can use wildcards when specifying network interface names. For example: `eth*`
Private network Interfaces	Oracle Clusterware, in the Grid Plug and Play (GPnP) Profile	Configure an interface for use as a private interface during installation by marking the interface as **Private**, or use the `oifcfg setif` command to designate an interface as a private interface. **See Also:** "OIFCFG Commands" on page D-2 for more information about the `oifcfg setif` command

Redundant Interconnect Usage

You can define multiple interfaces for Redundant Interconnect Usage by classifying the role of interfaces as private either during installation or after installation using the `oifcfg setif` command. When you do, Oracle Clusterware creates from one to four (depending on the number of interfaces you define) highly available IP (HAIP) addresses, which Oracle Database and Oracle ASM instances use to ensure highly available and load balanced communications.

The Oracle software (including Oracle RAC, Oracle ASM, and Oracle ACFS, all 11*g* release 2 (11.2.0.2), or later), by default, uses the HAIP address of the interfaces designated with the private role as the HAIP address for all of its traffic, enabling load balancing across the provided set of cluster interconnect interfaces. If one of the defined cluster interconnect interfaces fails or becomes non-communicative, then Oracle Clusterware transparently moves the corresponding HAIP address to one of the remaining functional interfaces.

For example, after installation, if you add a new interface to a server named `eth3` with the subnet number 172.16.2.0, then use the following command to make this interface available to Oracle Clusterware for use as a private interface:

```
$ oifcfg setif -global eth3/172.16.2.0:cluster_interconnect
```

While Oracle Clusterware brings up a HAIP address on `eth3` of 169.254.*.* (which is the reserved subnet for HAIP), and the database, Oracle ASM, and Oracle ACFS use that address for communication, Oracle Clusterware also uses the 172.16.2.0 address for its own communication.

Caution: Do not use OIFCFG to classify HAIP subnets (169.264.*.*). You can use OIFCFG to record the interface name, subnet, and type (public, cluster interconnect, or Oracle ASM) for Oracle Clusterware. However, you cannot use OIFCFG to modify the actual IP address for each interface.

Note: Oracle Clusterware uses at most four interfaces at any given point, regardless of the number of interfaces defined. If one of the interfaces fails, then the HAIP address moves to another one of the configured interfaces in the defined set.

When there is only a single HAIP address and multiple interfaces from which to select, the interface to which the HAIP address moves is no longer the original interface upon which it was configured. Oracle Clusterware selects the interface with the *lowest* numeric subnet to which to add the HAIP address.

See Also: *Oracle Grid Infrastructure Installation Guide* for your platform for information about defining interfaces

Consequences of Changing Interface Names Using OIFCFG

The consequences of changing interface names depend on which name you are changing, and whether you are also changing the IP address. In cases where you are only changing the interface names, the consequences are minor. If you change the name for the public interface that is stored in OCR, then you also must modify the node applications for the cluster. Therefore, you must stop the node applications for this change to take effect.

See Also: My Oracle Support (formerly Oracle*MetaLink*) note 276434.1 for more details about changing the node applications to use a new public interface name, available at the following URL:

https://metalink.oracle.com

Changing a Network Interface

You can change a network interface and its associated subnet address using the following procedure. You must perform this change on all nodes in the cluster.

This procedure changes the network interface and IP address on each node in the cluster used previously by Oracle Clusterware and Oracle Database.

Caution: The interface that the Oracle RAC (RDBMS) interconnect uses must be the same interface that Oracle Clusterware uses with the host name. Do not configure the private interconnect for Oracle RAC on a separate interface that is not monitored by Oracle Clusterware.

1. Ensure that Oracle Clusterware is running on all of the cluster nodes by running the following command:

```
$ olsnodes -s
```

The command returns output similar to the following, showing that Oracle Clusterware is running on all of the nodes in the cluster:

```
./olsnodes -s
myclustera Active
myclusterc Active
myclusterb Active
```

2. Ensure that the replacement interface is configured and operational in the operating system on all of the nodes. Use the `ifconfig` command (or `ipconfig` on Windows) for your platform. For example, on Linux, use:

```
$ /sbin/ifconfig..
```

3. Add the new interface to the cluster as follows, providing the name of the new interface and the subnet address, using the following command:

```
$ oifcfg setif -global if_name/subnet:cluster_interconnect
```

You can use wildcards with the interface name. For example, `oifcfg setif -global "eth*/192.168.0.0:cluster_interconnect` is valid syntax. However, be careful to avoid ambiguity with other addresses or masks used with other cluster interfaces. If you use wildcards, then you see a warning similar to the following:

```
eth*/192.168.0.0 global cluster_interconnect
PRIF-29: Warning: wildcard in network parameters can cause mismatch
among GPnP profile, OCR, and system
```

Note: Legacy network configuration does not support wildcards; thus wildcards are resolved using current node configuration at the time of the update.

See Also: Appendix D, "Oracle Interface Configuration Tool (OIFCFG) Command Reference" for more information about using OIFCFG commands

4. After the previous step completes, you can remove the former subnet, as follows, by providing the name and subnet address of the former interface:

```
oifcfg delif -global if_name/subnet
```

For example:

```
$ oifcfg delif -global eth1/10.10.0.0
```

Caution: This step should be performed only after a replacement interface is committed into the Grid Plug and Play configuration. Simple deletion of cluster interfaces without providing a valid replacement can result in invalid cluster configuration.

5. Verify the current configuration using the following command:

```
oifcfg getif
```

For example:

```
$ oifcfg getif
eth2 10.220.52.0 global cluster_interconnect
eth0 10.220.16.0 global public
```

6. Stop Oracle Clusterware on all nodes by running the following command as `root` on each node:

```
# crsctl stop crs
```

> **Note:** With cluster network configuration changes, the cluster must be fully stopped; do not use rolling stops and restarts.

7. When Oracle Clusterware stops, you can deconfigure the deleted network interface in the operating system using the `ifconfig` command. For example:

   ```
   $ ifconfig down
   ```

   At this point, the IP address from network interfaces for the old subnet is deconfigured from Oracle Clusterware. This command does not affect the configuration of the IP address on the operating system.

   You must update the operating system configuration changes, because changes made using `ifconfig` are not persistent.

8. Restart Oracle Clusterware by running the following command on each node in the cluster as the `root` user:

   ```
   # crsctl start crs
   ```

   The changes take effect when Oracle Clusterware restarts.

   If you use the `CLUSTER_INTERCONNECTS` initialization parameter, then you must update it to reflect the changes.

Creating a Network Using SRVCTL

Use the following procedure to create a network for a cluster member node, and to add application configuration information:

1. Log in as `root`.
2. Add a node application to the node, using the following syntax, where:

   ```
   srvctl add nodeapps -node node_name -address {vip |
      addr}/netmask[/if1[|if2|...]] [-pingtarget "ping_target_list"]
   ```

   In the preceding syntax:

   - *`node_name`* is the name of the node
   - *`vip`* is the VIP name or *`addr`* is the IP address
   - *`netmask`* is the netmask
   - *`if1`*`[|`*`if2`*`|...]` is a pipe-delimited list of interfaces bonded for use by the application
   - *`-ping_target_list`* is a comma-delimited list of IP addresses or host names to ping

> **Notes:**
>
> - Use the `-pingtarget` parameter when link status monitoring does not work as it does in a virtual machine environment.
> - Enter the `srvctl add nodeapps -help` command to review other syntax options.

In the following example of using `srvctl add nodeapps` to configure an IPv4 node application, the node name is node1, the netmask is 255.255.252.0, and the interface is eth0:

```
# srvctl add nodeapps -node node1 -address
node1-vip.mycluster.example.com/255.255.252.0/eth0
```

> **See Also:** *Oracle Real Application Clusters Administration and Deployment Guide* for more information about the SRVCTL commands used in this procedure

Changing Network Address Types Using SRVCTL

You can configure a network interface for either IPv4, IPv6, or both types of addresses on a given network. If you configure redundant network interfaces using a third-party technology, then Oracle does not support configuring one interface to support IPv4 addresses and the other to support IPv6 addresses. You must configure network interfaces of a redundant interface pair with the same IP address type. If you use the Oracle Clusterware Redundant Interconnect feature, then you must use IPv4 addresses for the interfaces.

All the nodes in the cluster must use the same IP protocol configuration. Either all the nodes use only IPv4, or all the nodes use only IPv6, or all the nodes use both IPv4 and IPv6. You cannot have some nodes in the cluster configured to support only IPv6 addresses, and other nodes in the cluster configured to support only IPv4 addresses.

The local listener listens on endpoints based on the address types of the subnets configured for the network resource. Possible types are IPV4, IPV6, or both.

Changing Static IPv4 Addresses To Static IPv6 Addresses Using SRVCTL

> **Note:** If the IPv4 network is in mixed mode with both static and dynamic addresses, then you cannot perform this procedure. You must first transition all addresses to static.

When you change from IPv4 static addresses to IPv6 static addresses, you add an IPv6 address and modify the network to briefly accept both IPv4 and IPv6 addresses, before switching to using static IPv6 addresses, only.

To change a static IPv4 address to a static IPv6 address:

1. Add an IPv6 subnet using the following command as `root` once for the entire network:

   ```
   # srvctl modify network -subnet ipv6_subnet/prefix_length
   ```

   In the preceding syntax *`ipv6_subnet/prefix_length`* is the subnet of the IPv6 address to which you are changing along with the prefix length, such as 3001::/64)

2. Add an IPv6 VIP using the following command as `root` once on each node:

   ```
   # srvctl modify vip -node node_name -netnum network_number -address vip_
   name/netmask
   ```

   In the preceding syntax:

   - *`node_name`* is the name of the node
   - *`network_number`* is the number of the network

- *vip_name*/*netmask* is the name of a local VIP that resolves to both IPv4 and IPv6 addresses

  The IPv4 netmask or IPv6 prefix length that follows the VIP name must satisfy two requirements:

  – If you specify a netmask in IPv4 format (such as 255.255.255.0), then the VIP name resolves to IPv4 addresses (but can also resolve to IPv6 addresses). Similarly, if you specify an IPv6 prefix length (such as 64), then the VIP name resolves to IPv6 addresses (but can also resolve to IPv4 addresses).

  – If you specify an IPv4 netmask, then it should match the netmask of the registered IPv4 network subnet number, regardless of whether the `-iptype` of the network is IPv6. Similarly, if you specify an IPv6 prefix length, then it must match the prefix length of the registered IPv6 network subnet number, regardless of whether the `-iptype` of the network is IPv4.

3. Add the IPv6 network resource to OCR using the following command:

```
oifcfg setif -global if_name/subnet:public
```

   See Also: "OIFCFG Command Format" on page D-2 for usage information for this command

4. Update the SCAN in DNS to have as many IPv6 addresses as there are IPv4 addresses. Add IPv6 addresses to the SCAN VIPs using the following command as `root` once for the entire network:

```
# srvctl modify scan -scanname scan_name
```

   scan_name is the name of a SCAN that resolves to both IPv4 and IPv6 addresses.

5. Convert the network IP type from IPv4 to both IPv4 and IPv6 using the following command as `root` once for the entire network:

```
srvctl modify network -netnum network_number -iptype both
```

   This command brings up the IPv6 static addresses.

6. Change all clients served by the cluster from IPv4 networks and addresses to IPv6 networks and addresses.

7. Transition the network from using both protocols to using only IPv6 using the following command:

```
# srvctl modify network -iptype ipv6
```

8. Modify the VIP using a VIP name that resolves to IPv6 by running the following command as `root`:

```
# srvctl modify vip -node node_name -address vip_name -netnum network_number
```

   Do this once for each node.

9. Modify the SCAN using a SCAN name that resolves to IPv6 by running the following command:

```
$ srvctl modify scan -scanname scan_name
```

   Do this once for the entire cluster.

> **See Also:** *Oracle Real Application Clusters Administration and Deployment Guide* for more information about the SRVCTL commands used in this procedure

Changing Dynamic IPv4 Addresses To Dynamic IPv6 Addresses Using SRVCTL

> **Note:** If the IPv4 network is in mixed mode with both static and dynamic addresses, then you cannot perform this procedure. You must first transition all addresses to dynamic.

To change a dynamic IPv4 address to a dynamic IPv6 address:

1. Add an IPv6 subnet using the `srvctl modify network` command.

   To add the IPv6 subnet, log in as `root` and use the following command syntax:

   ```
   srvctl modify network -netnum network_number -subnet ipv6_subnet/
      ipv6_prefix_length[/interface] -nettype autoconfig
   ```

   In the preceding syntax:

   - *`network_number`* is the number of the network
   - *`ipv6_subnet`* is the subnet of the IPv6 address to which you are changing (for example, 2001:db8:122:344:c0:2:2100::)
   - *`ipv6_prefix_length`* is the prefix specifying the IPv6 network address (for example, 64)

   For example, the following command modifies network `3` by adding an IPv6 subnet, 2001:db8:122:344:c0:2:2100::, and the prefix length 64:

   ```
   # srvctl modify network -netnum 3 -subnet
        2001:db8:122:344:c0:2:2100::/64 -nettype autoconfig
   ```

2. Add the IPv6 network resource to OCR using the following command:

   ```
   oifcfg setif -global if_name/subnet:public
   ```

   > **See Also:** "OIFCFG Command Format" on page D-2 for usage information for this command

3. Start the IPv6 dynamic addresses, as follows:

   ```
   srvctl modify network -netnum network_number -iptype both
   ```

   For example, on network number 3:

   ```
   # srvctl modify network -netnum 3 -iptype both
   ```

4. Change all clients served by the cluster from IPv4 networks and addresses to IPv6 networks and addresses.

   At this point, the SCAN in the GNS-delegated domain *`scan_name.gns_domain`* will resolve to three IPv4 and three IPv6 addresses.

5. Turn off the IPv4 part of the dynamic addresses on the cluster using the following command:

   ```
   # srvctl modify network -iptype ipv6
   ```

After you run the preceding command, the SCAN (*scan_name.gns_domain*) will resolve to only three IPv6 addresses.

Changing an IPv4 Network to an IPv4 and IPv6 Network

To change an IPv4 network to an IPv4 and IPv6 network, you must add an IPv6 network to an existing IPv4 network, as you do in steps 1 through 5 of the procedure documented in "Changing Static IPv4 Addresses To Static IPv6 Addresses Using SRVCTL" on page 2-26.

After you complete those three steps, log in as the Grid user, and run the following command:

```
$ srvctl status scan
```

Review the output to confirm the changes to the SCAN VIPs.

Transitioning from IPv4 to IPv6 Networks for VIP Addresses Using SRVCTL

Enter the following command to remove an IPv4 address type from a combined IPv4 and IPv6 network:

```
# srvctl modify network -iptype ipv6
```

This command starts the removal process of IPv4 addresses configured for the cluster.

3

Policy-Based Cluster and Capacity Management

This chapter provides an overview of Oracle Clusterware *policy-based management* of servers and resources used by Oracle databases or applications.

This chapter includes the following topics:

Overview of Server Pools and Policy-Based Management

Oracle Clusterware 11*g* release 2 (11.2) introduced **server pool**s, where **resource**s that Oracle Clusterware manages are contained in logical groups of servers called server pools. Resources are hosted on a shared infrastructure and are contained within server pools. Examples of resources that Oracle Clusterware manages are database instances, database services, application VIPs, and application components.

In an **Oracle Flex Cluster**, with **Hub Node**s and **Leaf Node**s, you can use server pools to run particular types of workloads on cluster member nodes, while providing simplified administration options. You can use a **cluster configuration policy set** to provide dynamic management of cluster policies across the cluster.

> **See Also :** Chapter 4, "Oracle Flex Clusters" for details about Oracle Flex Cluster configuration

You can continue to manage resources in an Oracle Clusterware standard Cluster by using the Oracle Clusterware 11*g* release 2 (11.2) server pool model, or you can manually manage resources by using the traditional fixed, non-server pool method.

This section includes the following topics:

Server Pools and Server Categorization

Administrators can deploy and manage servers dynamically using server pools by identifying servers distinguished by particular attributes, a process called **server categorization**. In this way, you can create clusters made up of heterogeneous nodes.

> **See Also:** "Overview of Server Categorization" on page 3-8 for details about server categorization

Server Pools and Policy-Based Management

With policy-based management, administrators specify the server pool (excluding the Generic and Free pools) in which the servers run. For example, a database administrator uses SRVCTL to create a server pool for servers hosting a database or database service. A clusterware administrator uses CRSCTL to create server pools for non-database use, such as creating a server pool for servers hosting an application.

Policy-based management:

- Enables online server reallocation based on a defined policy to satisfy workload capacity requirements
- Guarantees the allocation of required resources for critical work as defined by the policy
- Ensures isolation where necessary, so that you can provide dedicated servers in a cluster for applications and databases
- Enables policies to be configured to change pools in accordance with business needs or application demand, so that pools provide the required capacity at the right time

Server pools provide resource isolation to prevent applications running in one server pool from accessing resources running in another server pool. Oracle Clusterware provides fine-grained role separation between server pools. This capability maintains required management role separation between these groups in organizations that have clustered environments managed by separate groups.

> **See Also:** Appendix B, "Oracle Clusterware Resource Reference" for more information about resource attributes

Oracle Clusterware efficiently allocates servers in the cluster. Server pool attributes, defined when the server pool is created, dictate placement and prioritization of servers based on the `IMPORTANCE` server pool attribute.

> **See Also:** "Overview of Cluster Configuration Policies and the Policy Set" on page 3-9 for details about managing server pools to respond to business or application demand

How Server Pools Work

Server pools divide the cluster into logical groups of servers hosting both singleton and uniform applications. The application can be a database service or a non-database application. An application is *uniform* when the application workload is distributed over all servers in the server pool. An application is *singleton* when it runs on a single server within the server pool. Oracle Clusterware role-separated management determines access to and use of a server pool.

You manage server pools that contain Oracle RAC databases with the Server Control (SRVCTL) utility. Use the Oracle Clusterware Control (CRSCTL) utility to manage all other server pools. Only cluster administrators have permission to create top-level server pools.

Database administrators use the Server Control (SRVCTL) utility to create and manage server pools that will contain Oracle RAC databases. Cluster administrators use the Oracle Clusterware Control (CRSCTL) utility to create and manage all other server pools, such as server pools for non-database applications. Only cluster administrators have permission to create top-level server pools.

Top-level server pools:

- Logically divide the cluster
- Are always exclusive, meaning that one server can only reside in one particular server pool at a certain point in time

Default Server Pools

When Oracle Clusterware is installed, two internal server pools are created automatically: *Generic* and *Free*. All servers in a new installation are assigned to the Free server pool, initially. Servers move from Free to newly defined server pools automatically.

The Free Server Pool

The Free server pool contains servers that are not assigned to any other server pools. The attributes of the Free server pool are restricted, as follows:

- `SERVER_NAMES`, `MIN_SIZE`, and `MAX_SIZE` cannot be edited by the user
- `IMPORTANCE` and `ACL` can be edited by the user

The Generic Server Pool

The Generic server pool stores any server that is not in a top-level server pool and is not policy managed. Servers that host non-policy-managed applications, such as administrator-managed databases, are statically assigned to the Generic server pool.

The Generic server pool’s attributes are restricted, as follows:

- No one can modify configuration attributes of the Generic server pool (all attributes are read-only)
- You can only create administrator-managed databases in the Generic Pool, if the server you want to create the database on is one of the following:
  - Online and exists in the Generic server pool
  - Online and exists in the Free server pool, in which case Oracle Clusterware moves the server into the Generic server pool

- Online and exists in any other server pool *and* the user is either a cluster administrator or is allowed to use the server pool's servers, in which case, the server is moved into the Generic server pool
- Offline and the user is a cluster administrator

Server Pool Attributes

You can use SRVCTL or CRSCTL to create server pools for databases and other applications, respectively. If you use SRVCTL to create a server pool, then you can only use a subset of the server pool attributes described in this section. If you use CRSCTL to create server pools, then you can use the entire set of server pool attributes.

Server pool attributes are the attributes that you define to create and manage server pools.

The decision about which utility to use is based upon the type of resource being hosted in the server pool. You must use SRVCTL to create server pools that host Oracle databases. You must use CRSCTL to create server pools that host non-database resources such as middle tiers and applications.

When you use SRVCTL to create a server pool, the server pool attributes available to you are:

```
-category
-importance
-min
-max
-serverpool
-servers
```

SRVCTL prepends "ora." to the name of the server pool.

> **See Also:** *Oracle Real Application Clusters Administration and Deployment Guide* for more information

When you use CRSCTL to create a server pool, all attributes listed and described in Table 3–1 are available to you.

> **See Also:** `"crsctl add serverpool"` on page E-42 for more information

Table 3–1 Server Pool Attributes

Attribute	Values and Format	Description
`ACL`	String in the following format: `owner:`*`user`*`:rwx,pgrp:`*`group`*`:rwx,other::r–`	Defines the owner of the server pool and which privileges are granted to various operating system users and groups. The server pool owner defines the operating system user of the owner, and which privileges that user is granted. The value of this optional attribute is populated at the time a server pool is created based on the ACL of the user creating the server pool, unless explicitly overridden. The value can subsequently be changed, if such a change is allowed based on the existing privileges of the server pool. In the string: ■ `owner`: The operating system user of the server pool owner, followed by the privileges of the owner ■ `pgrp`: The operating system group that is the primary group of the owner of the server pool, followed by the privileges of members of the primary group ■ `other`: Followed by privileges of others ■ `r`: Read only ■ `w`: Modify attributes of the pool or delete it ■ `x`: Assign resources to this pool By default, the identity of the client that creates the server pool is the `owner`. Also by default, `root`, and the *`user`* specified in `owner` have full privileges. You can grant required operating system users and operating system groups their privileges by adding the following lines to the `ACL` attribute: The operating system user that creates the server pool is the owner of the server pool, and the `ACL` attribute for the server pool reflects the UNIX-like read, write, and execute ACL definitions for the user, primary group, group, and other.
`ACTIVE_SERVERS`	A string of server names in the following format: `server_name1 server_name2 ...`	Oracle Clusterware automatically manages this attribute, which contains the space-delimited list of servers that are currently assigned to a server pool.
`EXCLUSIVE_POOLS`	String	This optional attribute indicates if servers assigned to this server pool are shared with other server pools. A server pool can explicitly state that it is mutually exclusive of any other server pool that has the same value for this attribute. Two or more server pools are mutually exclusive when the sets of servers assigned to them do not have a single server in common. For example, server pools A and B must be mutually exclusive if they both have the value of this attribute set to the same string, such as `pools_A_B`. Top-level server pools are mutually exclusive, by default.
`IMPORTANCE`	Any integer from 0 to 1000	Relative importance of the server pool, with `0` denoting the lowest level of importance and `1000`, the highest. This optional attribute is used to determine how to reconfigure the server pools when a node joins or leaves the cluster. The default value is `0`.
`MIN_SIZE`	Any nonnegative integer	The minimum size of a server pool. If the number of servers contained in a server pool is below the number you specify in this attribute, then Oracle Clusterware automatically moves servers from other pools into this one until that number is met. **Note:** The value of this optional attribute does not set a hard limit. It governs the priority for server assignment whenever the cluster is reconfigured. The default value is `0`.

Table 3–1 (Cont.) Server Pool Attributes

Attribute	Values and Format	Description
MAX_SIZE	Any nonnegative integer or -1 (no limit)	The maximum number of servers a server pool can contain. This attribute is optional and is set to -1 (no limit), by default. **Note:** A value of -1 for this attribute spans the entire cluster.
NAME	String	The name of the server pool, which you must specify when you create the server pool. Server pool names must be unique within the domain of names of user-created entities, such as resources, types, and servers. A server pool name has a 254 character limit and can contain any platform-supported characters except the exclamation point (!), the tilde (~), and spaces. A server pool name cannot begin with a period nor with *ora*. **Note:** When you create server pools using SRVCTL, the utility prepends "ora." to the name of the server pool.
PARENT_ POOLS	A string of space-delimited server pool names in the following format: sp1 sp2 ...	Use of this attribute makes it possible to create nested server pools. Server pools listed in this attribute are referred to as *parent* server pools. A server pool included in a parent server pool is referred to as a *child* server pool. **Note:** If you use SRVCTL to create the server pool, then you cannot specify this attribute.
SERVER_ CATEGORY	String	The name of a registered server category, used as part of server categorization. Oracle Clusterware standard Clusters and Oracle Flex Clusters have default categories of hub and leaf. When you create a server pool, if you set a value for SERVER_CATEGORY, then you cannot set a value for SERVER_NAMES. Only one of these parameters may have a non-empty value. Use the SERVER_CATEGORY attribute to classify servers assigned to a server pool based on server attributes. You can organize servers and server pools in a cluster to match specific workload to servers and server pools, based on server attributes that you define. **See Also:** "crsctl status server" on page E-91 for a list of server attributes
SERVER_ NAMES	A string of space-delimited server names in the following format: server1 server2 ...	A list of candidate node names that may be associated with a server pool. If you do not provide a set of server names for this optional attribute, then Oracle Clusterware is configured so that any server may be assigned to any server pool, to the extent allowed by values of other attributes, such as PARENT_POOLS. The server names identified as candidate node names are not validated to confirm that they are currently active cluster members. Use this attribute to define servers as candidates that have not yet been added to the cluster. If you set a value for SERVER_NAMES, then you cannot set a value for SERVER_ CATEGORY; Only one of these attributes may have a non-empty value. **Note:** If you set the SERVER_CATEGORY attribute and you need to specify individual servers, then you can list servers by name using the EXPRESSION server category attribute.

How Oracle Clusterware Assigns New Servers Using Server Pools

Oracle Clusterware assigns new servers to server pools in the following order:

1. Generic server pool
2. User-created server pool
3. Free server pool

Oracle Clusterware continues to assign servers to server pools until the following conditions are met:

1. Until all server pools are filled in order of importance to their minimum (MIN_ SIZE).

2. Until all server pools are filled in order of importance to their maximum (MAX_SIZE).
3. By default, any servers not placed in a server pool go into the Free server pool.

   You can modify the IMPORTANCE attribute for the Free server pool. If the value of the IMPORTANCE attribute of the Free server pool is greater than one or more of the other server pools, then the Free server pool will receive any remaining servers once the value of their MIN_SIZE attribute is met.

When a server joins a cluster, several things occur.

Consider the server pools configured in Table 3–2:

Table 3–2 Sample Server Pool Attributes Configuration

NAME	IMPORTANCE	MIN_SIZE	MAX_SIZE	PARENT_ POOLS	EXCLUSIVE_ POOLS
sp1	1	1	10		
sp2	3	1	6		
sp3	2	1	2		
sp2_1	2	1	5	sp2	S123
sp2_2	1	1	5	sp2	S123

For example, assume that there are no servers in a cluster; all server pools are empty.

When a server, named server1, joins the cluster:

1. Server-to-pool assignment commences.
2. Oracle Clusterware only processes top-level server pools (those that have no parent server pools), first. In this example, the top-level server pools are sp1, sp2, and sp3.
3. Oracle Clusterware lists the server pools in order of IMPORTANCE, as follows: sp2, sp3, sp1.
4. Oracle Clusterware assigns server1 to sp2 because sp2 has the highest IMPORTANCE value and its MIN_SIZE value has not yet been met.
5. Oracle Clusterware processes the remaining two server pools, sp2_1 and sp2_2. The sizes of both server pools are below the value of the MIN_SIZE attribute (both server pools are empty and have MIN_SIZE values of 1).
6. Oracle Clusterware lists the two remaining pools in order of IMPORTANCE, as follows: sp2_1, sp2_2.
7. Oracle Clusterware assigns server1 to sp2_1 but cannot assign server1 to sp2_2 because sp2_1 is configured to be exclusive with sp2_2.

After processing, the cluster configuration appears, as follows

Table 3–3 Post Processing Server Pool Configuration

Server Pool Name	Assigned Servers
sp1	
sp2	server1
sp3	
sp2_1	server1

Table 3–3 (Cont.) Post Processing Server Pool Configuration

Server Pool Name	Assigned Servers
sp2_2	

Servers Moving from Server Pool to Server Pool

If the number of servers in a server pool falls below the value of the MIN_SIZE attribute for the server pool (such as when a server fails), based on values you set for the MIN_SIZE and IMPORTANCE attributes for all server pools, Oracle Clusterware can move servers from other server pools into the server pool whose number of servers has fallen below the value for MIN_SIZE. Oracle Clusterware selects servers from other server pools to move into the deficient server pool that meet the following criteria:

- For server pools that have a lower IMPORTANCE value than the deficient server pool, Oracle Clusterware can take servers from those server pools even if it means that the number of servers falls below the value for the MIN_SIZE attribute.
- For server pools with equal or greater IMPORTANCE, Oracle Clusterware only takes servers from those server pools if the number of servers in a server pool is greater than the value of its MIN_SIZE attribute.

Managing Server Pools Using Default Attributes

By default, each server pool is configured with the following attribute options for managing server pools:

- MIN_SIZE: The minimum number of servers the server pool should contain.

  If the number of servers in a server pool is below the value of this attribute, then Oracle Clusterware automatically moves servers from elsewhere into the server pool until the number of servers reaches the attribute value.
- MAX_SIZE: The maximum number of servers the server pool should contain.
- IMPORTANCE: A number from 0 to 1000 (0 being least important) that ranks a server pool among all other server pools in a cluster.

In addition, you can assign additional attributes to provide more granular management of server pools, as part of a cluster configuration policy. Attributes such as EXCLUSIVE_POOLS and SERVER_CATEGORY can assist you to create policies for your server pools that enhance performance and build tuning design management into your server pool.

Overview of Server Categorization

Oracle Clusterware 11*g* release 2 (11.2) introduced server pools as a means for specifying resource placement and administering server allocation and access. Originally, server pools were restricted to a set of basic attributes characterizing servers as belonging to a given pool, with no way to distinguish between *types* of servers; all servers were considered to be equal in relation to their processors, physical memory, and other characteristics.

Server categorization enables you to organize servers into particular categories by using attributes such as processor types, memory, and other distinguishing system features. You can configure server pools to restrict eligible members of the pool to a category of servers, which share a particular set of attributes.

Note: If you create policies with Oracle Database Quality of Service Management (Oracle Database QoS Management), then you categorize servers by setting server pool directive overrides, and CRSCTL commands using the `policy` and `policyset` nouns are disabled. Also if you switch from using Oracle Clusterware policies to using Oracle Database QoS Management policies, then the Oracle Clusterware policies are replaced, because the two policy types cannot coexist. Oracle recommends that you create a backup using `crsctl status policyset -file` *`file_name`* before you switch policies.

See Also:

- Appendix E, "Oracle Clusterware Control (CRSCTL) Utility Reference" for information about CRSCTL commands you can use for policy set management
- *Oracle Database Quality of Service Management User's Guide* for more information about server pool directive overrides

Overview of Cluster Configuration Policies and the Policy Set

A **cluster configuration policy** is a document that contains exactly one definition for each server pool managed by the cluster configuration policy set. A **cluster configuration policy set** is a document that defines the names of all server pools configured for the cluster and definitions for all policies.

Note: Oracle Clusterware 11*g* release 2 (11.2) supports only a single server pool configuration. You must manually make any changes to the server pool configuration when you want the change to take effect.

In Oracle Clusterware 12*c*, you use the policies defined in the cluster configuration policy set for server pool specification and management, and Oracle Clusterware manages the server pools according to the policies in the policy set. With a cluster configuration policy set, for example, you can allocate more servers to OLTP workload during weekly business hours to respond to email demand, and on the weekends and evenings, allocate more servers to batch workloads, and perform transitions of server pool configuration or server allocation, atomically.

At any point in time, only one policy is in effect for the cluster. But you can create several different policies, so that you can configure pools of servers with parameters to reflect differences in requirements for the cluster based on business needs or demand, or based on calendar dates or times of the day.

See Also: "An Example Policy Set Configuration" on page 3-15 for a more detailed example of policy set configuration

Server Configuration and Server State Attributes

Oracle Clusterware assigns each server a set of attributes as soon as you add a server to a cluster. Some of these attributes describe the physical characteristics of the server, while others describe the state conditions of the server. Also, there are other server attributes which you can modify that help further categorize servers. If you remove the server from the cluster, then Oracle Clusterware deletes the server object.

You use server configuration attributes to categorize servers, as part of a server categorization management policy.

Table 3–4 lists and describes server configuration attributes.

Table 3–4 Server Configuration Attributes

Attribute	Description
`ACTIVE_CSS_ROLE`	Role being performed by the server. A server can have one of the following roles: ■ `hub`: Designated role for a server in an Oracle Flex Cluster or the designated role for any node in an Oracle Clusterware standard Cluster. ■ `leaf`: The server is a Leaf Node in an Oracle Flex Cluster. **Note:** You cannot configure this attribute.
`CONFIGURED_CSS_ROLE`	Configured role for the server. A server can be either of the following: `hub`: Designated role for a server in an Oracle Flex Cluster or the designated role for any node in an Oracle Clusterware standard Cluster. `leaf`: The server is a Leaf Node in an Oracle Flex Cluster. **Note:** You cannot configure this attribute.
`CPU_CLOCK_RATE`	CPU clock rate in megahertz (MHz)
`CPU_COUNT`	Number of processors

`CPU_EQUIVALENCY`	The relative value (expressed as a positive integer greater than or equal to 1) that Oracle Clusterware uses to describe that the CPU power provided by a server may deviate (positively or negatively) from its physical representation using a baseline of 1000, for example. A value lower than 1000 describes that, despite a certain value for the `CPU_COUNT` and `CPU_CLOCK_RATE` parameters, the equivalent power provided by this server is respectively lower.
	Use the following commands to view or modify, respectively, this attribute on the local server:
	`crsctl get cpu equivalency`
	`crsctl set cpu equivalency`

`CPU_HYPERTHREADING`	Status of hyperthreading for the CPU. A value of `0` signifies that hyperthreading is not in use. A value of `1` signifies that hyperthreading is in use.
`MEMORY_SIZE`	Memory size in megabytes (MB)
`NAME`	The name of the server.
`RESOURCE_USE_ENABLED`	A server pool resource management parameter. If the value for this attribute is 1, which is the default, then the server can be used for resource placement. If the value is 0, then Oracle Clusterware disallows starting server pool resources on the server. The server remains in the Free pool. You can review the setting and control this attribute for each cluster member node by using the `crsctl get resource use` and `crsctl set resource use` commands.

`SERVER_LABEL`	An arbitrary value that you can use to label the server. You can use this attribute when setting up server categories. For example, you can specify a location (such as building_A or building_B), which makes it possible to put servers into pools where location is a requirement, by creating an appropriate server category and using it for the server pool.
	Use the following commands to view or modify, respectively, this attribute on the local server:
	`crsctl get server label`
	`crsctl set server label`

Table 3–5 lists and describes read-only server state and configuration attributes:

Table 3–5 Server State Attributes

Attribute	Description
ACTIVE_POOLS	A space-delimited list of the names of the server pools to which a server belongs. Oracle Clusterware manages this list automatically.
STATE	A server can be in one of the following states: ■ ONLINE: The server is a member of the cluster and is available for resource placement. ■ OFFLINE: The server is not currently a member of the cluster. Subsequently, it is not available for resource placement. ■ JOINING: When a server joins a cluster, Oracle Clusterware processes the server to ensure that it is valid for resource placement. Oracle Clusterware also checks the state of resources configured to run on the server. Once the validity of the server and the state of the resources are determined, the server transitions out of this state. ■ LEAVING: When a planned shutdown for a server begins, the state of the server transitions to LEAVING, making it unavailable for resource placement. ■ VISIBLE: Servers that have Oracle Clusterware running, but not the Cluster Ready Services daemon (CRSD), are put into the VISIBLE state. This usually indicates an intermittent issue or failure and Oracle Clusterware trying to recover (restart) the daemon. Oracle Clusterware cannot manage resources on servers while the servers are in this state. ■ RECONFIGURING: When servers move between server pools due to server pool reconfiguration, a server is placed into this state if resources that ran on it in the current server pool must be stopped and relocated. This happens because resources running on the server may not be configured to run in the server pool to which the server is moving. As soon as the resources are successfully relocated, the server is put back into the ONLINE state. Use the crsctl status server command to obtain server information.

Table 3–5 (Cont.) Server State Attributes

Attribute	Description
`STATE_DETAILS`	This is a read-only attribute that Oracle Clusterware manages. The attribute provides additional details about the state of a server. Possible additional details about a server state are: Server state: `ONLINE`: ■ `AUTOSTARTING RESOURCES` Indicates that the resource autostart procedure (performed when a server reboots or the Oracle Clusterware stack is restarted) is in progress for the server. ■ `AUTOSTART QUEUED` The server is waiting for the resource autostart to commence. Once that happens, the attribute value changes to `AUTOSTARTING RESOURCES`. Server state: `RECONFIGURING`: ■ `STOPPING RESOURCES` Resources that are restricted from running in a new server pool are stopping. ■ `STARTING RESOURCES` Resources that can run in a new server pool are starting. ■ `RECONFIG FAILED` One or more resources did not stop and thus the server cannot transition into the `ONLINE` state. At this point, manual intervention is required. You must stop or unregister resources that did not stop. After that, the server automatically transitions into the `ONLINE` state. Server state: `JOINING`: ■ `CHECKING RESOURCES` Whenever a server reboots, the Oracle Clusterware stack restarts, or CRSD on a server restarts, the policy engine must determine the current state of the resources on the server. While that procedure is in progress, this value is returned.

Managing Memory Pressure for Database Servers

Enterprise database servers can use all available memory due to too many open sessions or runaway workloads. Running out of memory can result in failed transactions or, in extreme cases, a restart of the server and the loss of a valuable resource for your applications. Oracle Database QoS Management detects memory pressure on a server in real time and redirects new sessions to other servers to prevent using all available memory on the stressed server.

When Oracle Database QoS Management is enabled and managing an Oracle Clusterware server pool, Cluster Health Monitor sends a metrics stream that provides real-time information about memory resources for the cluster servers to Oracle Database QoS Management. This information includes the following:

- Amount of available memory
- Amount of memory currently in use

If Oracle Database QoS Management determines that a node is under memory stress, then the database services managed by Oracle Clusterware are stopped on that node, preventing new connections from being created. After the memory stress is relieved, the services on that node are restarted automatically, and the listener starts sending new connections to that server. Memory pressure can be relieved in several ways (for example, by closing existing sessions or by user intervention).

Rerouting new sessions to different servers protects the existing workloads on the memory-stressed server and enables the server to remain available. Managing the memory pressure for servers adds a new resource protection capability in managing service levels for applications hosted on Oracle RAC databases.

Server Category Attributes

You define servers into named categories, and assign attributes that define servers as members of that category. Some attributes that you can use to define members of a category describe the state conditions for the server, and others describe the physical characteristics of the server. You can also create your own characteristics to define servers as members of a particular category.

Note: If you change the value of any of the server attributes listed in the `EXPRESSION` server category attribute, then you must restart the Oracle Clusterware technology stack on the affected servers before the new values take effect.

Table 3–6 lists and describes possible server category attributes.

Table 3–6 Server Category Attributes

Attribute	Values and Format	Description
`NAME`	String	The name of the server category, which you must specify when you create the server category. Server category names must be unique within the domain of names of user-created entities, such as resources, types, and servers. A server pool name has a 254 character limit and can contain any platform-supported characters except the exclamation point (!) and the tilde (~). A server pool name cannot begin with a period nor with *ora*.
`ACTIVE_CSS_ROLE`	`hub`, `leaf`	Active role for the server, which can be either of the following: `hub`: The server is a Hub Node in either an Oracle Flex Cluster or an Oracle Clusterware standard Cluster. This is the default value in either case. `leaf`: The server is a Leaf Node in an Oracle Flex Cluster.
`EXPRESSION`	String in the following format: (*`expression`*)	A set of server attribute names, values, and conditions that can be evaluated for each server to determine whether it belongs to the category. Table 3–4 lists and describes server attributes. Acceptable comparison operators include: **=**: equal **eqi**: equal, case insensitive **>**: greater than **<**: less than **!=**: not equal **co**: contains **coi**: contains, case insensitive **st**: starts with **en**: ends with **nc**: does not contain **nci**: does not contain, case insensitive Acceptable boolean operators include: **AND** **OR** **Note:** Spaces must surround the operators used in the `EXPRESSION` string. For example: `EXPRESSION='((NAME = server1) OR (NAME = server2))'"`

An Example Policy Set Configuration

Assume that you have a four-node cluster that is used by three different applications, *app1*, *app2*, and *app3*, and that you have created three server pools, *pool1*, *pool2*, and *pool3*. You configure the server pools such that each application is assigned to run in its own server pool, and that *app1* wants to have two servers, and *app2* and *app3* each want one server. The server pool configurations are as follows:

```
$ crsctl status serverpool pool1 -p
NAME=pool1
IMPORTANCE=0
MIN_SIZE=2
MAX_SIZE=2
SERVER_NAMES=
PARENT_POOLS=
EXCLUSIVE_POOLS=
ACL=owner:mjk:rwx,pgrp:g900:rwx,other::r--
SERVER_CATEGORY=
$ crsctl status serverpool pool2 -p
NAME=pool2
IMPORTANCE=0
MIN_SIZE=1
MAX_SIZE=1
SERVER_NAMES=
PARENT_POOLS=
EXCLUSIVE_POOLS=
ACL=owner:mjk:rwx,pgrp:g900:rwx,other::r--
SERVER_CATEGORY=

$ crsctl status serverpool pool3 -p
NAME=pool3
IMPORTANCE=0
MIN_SIZE=1
MAX_SIZE=1
SERVER_NAMES=
PARENT_POOLS=
EXCLUSIVE_POOLS=
ACL=owner:mjk:rwx,pgrp:g900:rwx,other::r--
SERVER_CATEGORY=
```

> **Notes:** The `crsctl status serverpool` command shown in the preceding examples only functions if you created the server pools using CRSCTL.

This configuration, however, does not consider the fact that some applications need server time at different times of the day, week, or month. Email applications, for example, typically use more resources during business hours and use less resources at night and on weekends.

Further assume that *app1* requires two servers during business hours, but only requires one server at night and does not require any servers on weekends. At the same time, *app2* and *app3* each require one server during business hours, while at night, *app2* requires two servers and *app3* requires one. On the weekend, *app2* requires one server and *app3* requires three. This scenario suggests three configurations that you must configure for the cluster:

1. **Day Time**:

   app1 uses two servers
   app2 and *app3* use one server, each

2. **Night Time**:

   app1 uses one server
   app2 uses two servers
   app3 uses one server

3. **Weekend**:

   app1 is not running (0 servers)
   app2 uses one server
   app3 uses three servers

Policy Set Creation

Given these assumptions, run the `crsctl create policyset` command to create a policy set with a single policy named `Default`, which reflects the configuration displayed by the `crsctl status serverpool` command. You can use the `Default` policy to create other policies to meet the needs assumed in this example. The `crsctl create policyset` command creates a text file similar to Example 3–1.

Example 3–1 Policy Set Text File

```
SERVER_POOL_NAMES=Free pool1 pool2 pool3
POLICY
  NAME=Default
  SERVERPOOL
    NAME=pool1
    IMPORTANCE=0
    MAX_SIZE=2
    MIN_SIZE=2
    SERVER_CATEGORY=
  SERVERPOOL
    NAME=pool2
    IMPORTANCE=0
    MAX_SIZE=1
    MIN_SIZE=1
    SERVER_CATEGORY=
  SERVERPOOL
    NAME=pool3
    IMPORTANCE=0
    MAX_SIZE=1
    MIN_SIZE=1
    SERVER_CATEGORY=
```

Policy Modification

To modify the preceding policy set to meet the needs assumed in this example, edit the text file to define policies for the three scenarios discussed previously, by changing the name of the policy from `Default` to `DayTime`. Then, copy the policy and paste it twice to form two subsequent policies, which you name `NightTime` and `Weekend`, as shown in Example 3–2.

Example 3–2 Modified Policy Set Text File

```
SERVER_POOL_NAMES=Free pool1 pool2 pool3
POLICY
  NAME=DayTime
  SERVERPOOL
    NAME=pool1
    IMPORTANCE=0
    MAX_SIZE=2
    MIN_SIZE=2
    SERVER_CATEGORY=
  SERVERPOOL
    NAME=pool2
    IMPORTANCE=0
    MAX_SIZE=1
    MIN_SIZE=1
    SERVER_CATEGORY=
  SERVERPOOL
    NAME=pool3
    IMPORTANCE=0
    MAX_SIZE=1
    MIN_SIZE=1
    SERVER_CATEGORY=
POLICY
```

```
  NAME=NightTime
  SERVERPOOL
    NAME=pool1
    IMPORTANCE=0
    MAX_SIZE=1
    MIN_SIZE=1
    SERVER_CATEGORY=
  SERVERPOOL
    NAME=pool2
    IMPORTANCE=0
    MAX_SIZE=2
    MIN_SIZE=2
    SERVER_CATEGORY=
  SERVERPOOL
    NAME=pool3
    IMPORTANCE=0
    MAX_SIZE=1
    MIN_SIZE=1
    SERVER_CATEGORY=
POLICY
  NAME=Weekend
  SERVERPOOL
    NAME=pool1
    IMPORTANCE=0
    MAX_SIZE=0
    MIN_SIZE=0
    SERVER_CATEGORY=
  SERVERPOOL
    NAME=pool2
    IMPORTANCE=0
    MAX_SIZE=1
    MIN_SIZE=1
    SERVER_CATEGORY=
  SERVERPOOL
    NAME=pool3
    IMPORTANCE=0
    MAX_SIZE=3
    MIN_SIZE=3
    SERVER_CATEGORY=
```

Notice that, in addition to changing the names of the individual policies, the `MAX_SIZE` and `MIN_SIZE` policy attributes for each of the server pools in each of the policies were also modified according to the needs of the applications.

The following command registers the policy set stored in a file with Oracle Clusterware:

```
$ crsctl modify policyset -file file_name
```

You can achieve the same results as shown in the previous examples by editing the `Default` policy set, as a whole, using the `crsctl modify policyset` command, and by using the `crsctl modify serverpool` command to change individual server pool attributes for a specific policy.

The following command modifies the `Default` policy set to manage the three server pools:

```
$ crsctl modify policyset –attr "SERVER_POOL_NAMES=Free pool1 pool2 pool3"
```

The following commands add the three policies:

```
$ crsctl add policy DayTime
$ crsctl add policy NightTime
$ crsctl add policy Weekend
```

The following commands configure the three server pools according to the requirements of the policies:

```
$ crsctl modify serverpool pool1 -attr "MIN_SIZE=2,MAX_SIZE=2" -policy DayTime
$ crsctl modify serverpool pool1 -attr "MIN_SIZE=1,MAX_SIZE=1" -policy NightTime
$ crsctl modify serverpool pool1 -attr "MIN_SIZE=0,MAX_SIZE=0" -policy Weekend

$ crsctl modify serverpool pool2 -attr "MIN_SIZE=1,MAX_SIZE=1" -policy DayTime
$ crsctl modify serverpool pool2 -attr "MIN_SIZE=2,MAX_SIZE=2" -policy NightTime
$ crsctl modify serverpool pool2 -attr "MIN_SIZE=1,MAX_SIZE=1" -policy Weekend

$ crsctl modify serverpool pool3 -attr "MIN_SIZE=1,MAX_SIZE=1" -policy DayTime
$ crsctl modify serverpool pool3 -attr "MIN_SIZE=1,MAX_SIZE=1" -policy NightTime
$ crsctl modify serverpool pool3 -attr "MIN_SIZE=3,MAX_SIZE=3" -policy Weekend
```

There are now three distinct policies to manage the server pools to accommodate the requirements of the three applications.

Policy Activation

The policy set is now configured and controlling the three server pools with three different policies. You can activate policies when necessary, prompting Oracle Clusterware to reconfigure a server pool according to each policy's configuration.

The following command activates the `DayTime` policy:

```
$ crsctl modify policyset -attr "LAST_ACTIVATED_POLICY=DayTime"
```

The current status of the resources is as follows:

```
$ crsctl status resource -t
--------------------------------------------------------------------------------
Name           Target  State        Server                   State details
--------------------------------------------------------------------------------
Cluster Resources
--------------------------------------------------------------------------------
app1
      1        ONLINE  ONLINE       mjk_has3_2               STABLE
      2        ONLINE  ONLINE       mjk_has3_0               STABLE
app2
      1        ONLINE  ONLINE       mjk_has3_1               STABLE
app3
      1        ONLINE  ONLINE       mjk_has3_3               STABLE
```

The status of the server pools is as follows:

```
$ crsctl stat serverpool
NAME=Free
ACTIVE_SERVERS=

NAME=Generic
ACTIVE_SERVERS=

NAME=pool1
ACTIVE_SERVERS=mjk_has3_0 mjk_has3_2

NAME=pool2
ACTIVE_SERVERS=mjk_has3_1
```

```
NAME=pool3
ACTIVE_SERVERS=mjk_has3_3
```

The servers are allocated according to the `DayTime` policy and the applications run on their respective servers.

The following command activates the `Weekend` policy (remember, because the server pools have different sizes, as servers move between server pools, some applications will be stopped and others will be started):

```
$ crsctl modify policyset -attr "LAST_ACTIVATED_POLICY=Weekend"
CRS-2673: Attempting to stop 'app1' on 'mjk_has3_2'
CRS-2673: Attempting to stop 'app1' on 'mjk_has3_0'
CRS-2677: Stop of 'app1' on 'mjk_has3_0' succeeded
CRS-2672: Attempting to start 'app3' on 'mjk_has3_0'
CRS-2677: Stop of 'app1' on 'mjk_has3_2' succeeded
CRS-2672: Attempting to start 'app3' on 'mjk_has3_2'
CRS-2676: Start of 'app3' on 'mjk_has3_2' succeeded
CRS-2676: Start of 'app3' on 'mjk_has3_0' succeeded
```

The current status of the resources is as follows:

```
$ crsctl status resource -t
--------------------------------------------------------------------------------
Name           Target  State        Server                   State details
--------------------------------------------------------------------------------
Cluster Resources
--------------------------------------------------------------------------------
app1
      1        ONLINE  OFFLINE                               STABLE
      2        ONLINE  OFFLINE                               STABLE
app2
      1        ONLINE  ONLINE       mjk_has3_1               STABLE
app3
      1        ONLINE  ONLINE       mjk_has3_0               STABLE
      2        ONLINE  ONLINE       mjk_has3_2               STABLE
      3        ONLINE  ONLINE       mjk_has3_3               STABLE
--------------------------------------------------------------------------------
```

The status of the server pools is as follows:

```
$ crsctl status serverpool
NAME=Free
ACTIVE_SERVERS=

NAME=Generic
ACTIVE_SERVERS=

NAME=pool1
ACTIVE_SERVERS=

NAME=pool2
ACTIVE_SERVERS=mjk_has3_1

NAME=pool3
ACTIVE_SERVERS=mjk_has3_0 mjk_has3_2 mjk_has3_3
```

Using the `crsctl modify policyset` command, Oracle Clusterware changed server pool configuration, moved servers according to the requirements of the policy, and stopped and started the applications.

See Also: Appendix E, "Oracle Clusterware Control (CRSCTL) Utility Reference" for complete details on using the CRSCTL commands shown in this example

4

Oracle Flex Clusters

An **Oracle Flex Cluster** scales Oracle Clusterware to large numbers of nodes.

This chapter includes the following topics:

- Overview of Oracle Flex Clusters
- Managing Oracle Flex Clusters

Overview of Oracle Flex Clusters

Oracle Grid Infrastructure installed in an Oracle Flex Cluster configuration is a scalable, dynamic, robust network of nodes. Oracle Flex Clusters provide a platform for a variety of applications, including Oracle Real Application Clusters (Oracle RAC) databases with large numbers of nodes. Oracle Flex Clusters also provide a platform for other service deployments that require coordination and automation for high availability.

All nodes in an Oracle Flex Cluster belong to a single Oracle Grid Infrastructure cluster. This architecture centralizes policy decisions for deployment of resources based on application needs, to account for various service levels, loads, failure responses, and recovery.

Oracle Flex Clusters contain two types of nodes arranged in a hub and spoke architecture: **Hub Node**s and **Leaf Node**s. The number of Hub Nodes in an Oracle Flex Cluster can be as many as 64. The number of Leaf Nodes can be many more. Hub Nodes and Leaf Nodes can host different types of applications.

Hub Nodes are similar to Oracle Grid Infrastructure nodes in an Oracle Clusterware standard Cluster configuration: they are tightly connected, and have direct access to shared storage. In an Oracle Flex Cluster configuration, shared storage can be provisioned to Leaf Nodes independent of the Oracle Grid Infrastructure.

Leaf Nodes are different from standard Oracle Grid Infrastructure nodes, in that they do not require direct access to shared storage, but instead request data through Hub Nodes. Hub Nodes can run in an Oracle Flex Cluster configuration without having any Leaf Nodes as cluster member nodes, but Leaf Nodes must be members of a cluster that includes at least one Hub Node.

> **Note:** If you upgrade an Oracle Flex Cluster, then Oracle recommends that you upgrade the Hub Nodes first, and that you also have any upgraded Hub Nodes up and running as part of the upgrade process.

Managing Oracle Flex Clusters

This section discusses Oracle Flex Cluster administration after successful installation of Oracle Grid Infrastructure for either a small or large cluster. Use CRSCTL to manage Oracle Flex Clusters.

See Also:

- *Oracle Grid Infrastructure Installation Guide* for more information about installing Oracle Grid Infrastructure
- Appendix E, "Oracle Clusterware Control (CRSCTL) Utility Reference" for more information about the various CRSCTL commands you use to manage Oracle Flex Clusters

This section includes the following topics:

- Changing the Cluster Mode
- Changing the Node Role

Changing the Cluster Mode

You can change the mode of an existing Oracle Clusterware standard Cluster to be an Oracle Flex Cluster.

Notes:

- Changing the cluster mode requires cluster downtime.
- Oracle does not support changing an Oracle Flex Cluster to an Oracle Clusterware standard Cluster.
- Oracle Flex Cluster requires Grid Naming Service (GNS).
- Zone delegation is not required.

Changing an Oracle Clusterware Standard Cluster to an Oracle Flex Cluster

To change an existing Oracle Clusterware standard Cluster to an Oracle Flex Cluster:

1. Run the following command to determine the current mode of the cluster:

   ```
   $ crsctl get cluster mode status
   ```

2. Run the following command to ensure that the Grid Naming Service (GNS) is configured with a fixed VIP:

   ```
   $ srvctl config gns
   ```

   This procedure cannot succeed unless GNS is configured with a fixed VIP. If there is no GNS, then, as `root`, create one, as follows:

   ```
   # srvctl add gns -vip vip_name | ip_address
   ```

   Run the following command as `root` to start GNS:

   ```
   # srvctl start gns
   ```

3. Use the Oracle Automatic Storage Management Configuration Assistant (ASMCA) to enable Oracle Flex ASM in the cluster before you change the cluster mode.

See Also: *Oracle Automatic Storage Management Administrator's Guide* for more information about enabling Oracle Flex ASM

4. Run the following command as `root` to change the mode of the cluster to be an Oracle Flex Cluster:

```
# crsctl set cluster mode flex
```

5. Stop Oracle Clusterware by running the following command as `root` on each node in the cluster:

```
# crsctl stop crs
```

6. Start Oracle Clusterware by running the following command as `root` on each node in the cluster:

```
# crsctl start crs -wait
```

Note: Use the `-wait` option to display progress and status messages.

Changing the Node Role

The configured role of a node, whether it is a Hub Node or a Leaf Node, is kept locally, and is initially set at the time of installation. At startup, a node tries to come up in whatever role it was last configured.

Use CRSCTL to change the role of a node, as follows:

1. Run the following command to determine the current role of the local node:

```
$ crsctl get node role config
```

2. Run the following command as `root` to change the role of the local node:

```
# crsctl set node role {hub | leaf}
```

Note: If you are changing a Leaf Node to a Hub Node, then you may have to run `srvctl add vip` to add a VIP, if a VIP does not already exist on the node. Leaf Nodes are not required to have VIPs.

If you installed the cluster with DHCP-assigned VIPs, then there is no need to manually add a VIP.

3. As `root`, stop Oracle High Availability Services on the node where you changed the role, as follows:

```
# crsctl stop crs
```

4. If you are changing a Leaf Node to a Hub Node, then configure the Oracle ASM Filter Driver as `root`, as follows:

```
# $ORACLE_HOME/bin/asmcmd afd_configure
```

See Also: *Oracle Automatic Storage Management Administrator's Guide* for more information about the `asmcmd afd_configure` command

5. As `root`, restart Oracle High Availability Services on the node where you changed the role, as follows:

```
# crsctl start crs -wait
```

> **Note:** Use the `-wait` option to display progress and status messages.

6. Perform steps 3 and 5 on the local node.
7. Manually update the inventory.

   If you convert a Hub Node to a Leaf Node, then run the following command on all remaining Hub Nodes:

```
$ Grid_home/oui/bin/runInstaller -updateNodeList ORACLE_HOME=Oracle_home
"CLUSTER_NODES={comma_separated_Hub_Node_list}" -silent -local CRS=TRUE
```

   On the newly converted Leaf Node, run the following command:

```
$ Grid_home/oui/bin/runInstaller -updateNodeList ORACLE_HOME=Oracle_home
"CLUSTER_NODES={Leaf_Node_name}" -silent -local CRS=TRUE
```

   If you convert a Leaf Node to a Hub Node, then run the following command on all Hub Nodes:

```
$ Grid_home/oui/bin/runInstaller -updateNodeList ORACLE_HOME=Oracle_home
"CLUSTER_NODES={comma_separated_Hub_Node_list}" -silent -local CRS=TRUE
```

> **See Also:** "Oracle RAC Environment CRSCTL Commands" on page E-36 for usage information about the preceding CRSCTL commands

5
Rapid Home Provisioning

Rapid Home Provisioning is a method of deploying software homes to nodes in a cloud computing environment from a single cluster where you create, store, and manage templates of Oracle homes as **image**s (called **gold image**s) of Oracle software, such as databases, middleware, and applications. You can make a **working copy** of any gold image, and then you can provision that working copy to any node in the cloud.

You store the gold images in a repository located on a **Rapid Home Provisioning Server**, which runs on one server in the Rapid Home Provisioning Server cluster that is a highly available provisioning system. This provisioning method simplifies quick patching of Oracle homes, thereby minimizing or completely eliminating downtime.

Also, using Rapid Home Provisioning, you can provision Oracle Database software for the various versions of the Oracle Database releases, such as 10.2.0.5 (Oracle Database 10*g*), 11.2.0.x (Oracle Database 11*g*), 12.1.0.1, and 12.1.0.2 (Oracle Database 12*c*). When you provision such software, Rapid Home Provisioning offers additional features for creating various types of databases (such as Oracle Real Application Clusters (Oracle RAC), single instance, and Oracle Real Application Clusters One Node (Oracle RAC One Node) databases) on different types of storage, and other options, such as using templates and creating container databases (CDBs). Rapid Home Provisioning improves and makes more efficient patching of database software, allowing for rapid and remote patching of the software, in most cases, without any downtime for the database.

> **See Also:** *Oracle Real Application Clusters Installation Guide* for your platform for more information about provisioning Oracle Database software

> **Notes:**
>
> - Oracle does not support Rapid Home Provisioning on Windows operating systems.
> - In Oracle Database 12*c* release 1 (12.1), the Rapid Home Provisioning Server does not manage operating system images or Oracle Grid Infrastructure homes.

Rapid Home Provisioning simplifies maintenance in large environments because you have only to update software homes one time on the Rapid Home Provisioning Server.

This section includes the following topics:

- Rapid Home Provisioning Architecture

- Implementing Rapid Home Provisioning
- Managing Rapid Home Provisioning Clients

Rapid Home Provisioning Architecture

The Rapid Home Provisioning architecture consists of Rapid Home Provisioning Servers and Rapid Home Provisioning Clients, and there is a Rapid Home Provisioning Server cluster and a Rapid Home Provisioning cluster.

The Rapid Home Provisioning Server cluster is a repository for all data, of which there are primarily two types:

- Gold images
- Metadata related to users, roles, permissions, and identities

The Rapid Home Provisioning Server acts as a central server for provisioning Oracle homes, making them available on Rapid Home Provisioning Client clusters.

Users operate on the Rapid Home Provisioning Server or Rapid Home Provisioning Client cluster site to request deployment of Oracle homes or to query gold images. When a user makes a request for an Oracle home, specifying a gold image, the Rapid Home Provisioning Client communicates with the Rapid Home Provisioning Server to pass on the request. The Rapid Home Provisioning Server processes the request by taking appropriate action to instantiate a copy of the gold image, and to make it available to the Rapid Home Provisioning Client cluster using available technologies such as Oracle Automatic Storage Management Cluster File System (Oracle ACFS), network file systems (NFSs), and snapshots.

This section includes the following topics:

- Rapid Home Provisioning Server
- Rapid Home Provisioning Client
- Rapid Home Provisioning Roles
- Rapid Home Provisioning Images

Rapid Home Provisioning Server

The Rapid Home Provisioning Server is a highly available software provisioning system that uses Oracle Automatic Storage Management (Oracle ASM), Oracle Automatic Storage Management Cluster File System (Oracle ACFS), Grid Naming Service (GNS), and other components. The Rapid Home Provisioning Server primarily acts as a central server for provisioning Oracle homes, making them available to Rapid Home Provisioning Clients.

Features of the Rapid Home Provisioning Server:

- Efficiently stores gold images for the managed homes, including separate binaries, and metadata related to users, roles, and permissions.
- Provides highly available network file system (HANFS) exports for homes accessed using mounts on remote clusters.
- Provides a list of available homes to clients upon request.
- Patch a software home once and then deploy the home to any Rapid Home Provisioning Client, instead of patching every site.
- Provides the ability to report on existing deployments.

Rapid Home Provisioning Client

The Rapid Home Provisioning Client is part of the Oracle Grid Infrastructure installed on all servers in the cloud. Users operate on a Rapid Home Provisioning Client to perform tasks such as requesting deployment of Oracle homes and listing available gold images. When a user requests an Oracle home specifying a gold image, the Rapid Home Provisioning Client communicates with the Rapid Home Provisioning Server to pass on the request. The Rapid Home Provisioning Server processes the request by instantiating a working copy of the gold image and making it available to the Rapid Home Provisioning Client using Oracle ACFS (recommended), a different local file system, or through NFS.

The Rapid Home Provisioning Client:

- Utilizes Oracle ACFS to store working copies which can be rapidly provisioned as local homes; new homes can be quickly created or undone using Oracle ACFS snapshots.

> **Note:** Oracle supports using other local file systems besides Oracle ACFS.

- Provides a list of available homes from the Rapid Home Provisioning Server.
- Allows high availability NFS (HANFS) mounts from the Rapid Home Provisioning Server to be provisioned as working copies on the Rapid Home Provisioning Client cluster.

The NFS home client is a Rapid Home Provisioning Client that does not have Oracle ACFS installed, but can still access the Rapid Home Provisioning Server through an NFS protocol. In an NFS protocol environment, the Rapid Home Provisioning Server must have a highly available NFS server configured so that the Rapid Home Provisioning Client will be highly available.

> **See Also:** "Creating a Rapid Home Provisioning Client" on page 5-9 for more information

Rapid Home Provisioning Roles

An administrator assigns roles to gold image users with access-level permissions defined for each role. Users on Rapid Home Provisioning Clients are also assigned specific roles. Rapid Home Provisioning includes basic built-in and composite built-in roles.

Basic Built-In Roles

The basic built-in roles and their functions are:

- GH_ROLE_ADMIN: An administrative role for everything related to roles. Users assigned this role are able to run `rhpctl` *`verb`* `role` commands.
- GH_SITE_ADMIN: An administrative role for everything related to Rapid Home Provisioning Clients. Users assigned this role are able to run `rhpctl` *`verb`* `client` commands.
- GH_SERIES_ADMIN: An administrative role for everything related to image series. Users assigned this role are able to run `rhpctl` *`verb`* `series` commands.

- GH_SERIES_CONTRIB: Users assigned this role can add images to a series using the `rhpctl insertimage series` command, or delete images from a series using the `rhpctl deleteimage series` command.
- GH_WC_ADMIN: An administrative role for everything related to working copies. Users assigned this role are able to run `rhpctl` *`verb`* `workingcopy` commands.
- GH_WC_OPER: A role that enables users to create a working copy for themselves or others using the `rhpctl add workingcopy` command with the `-user` option (when creating for others). Users assigned this role do not have administrative privileges and can only administer the working copies that they create.
- GH_WC_USER: A role that enables users to create a working copy using the `rhpctl add workingcopy` command. Users assigned this role do not have administrative privileges and can only delete working copies that they create.
- GH_IMG_ADMIN: An administrative role for everything related to images. Users assigned this are role are able to run `rhpctl` *`verb`* `image` commands.
- GH_IMG_USER: A role that enables users to create an image using the `rhpctl add | import image`. Users assigned this role do not have administrative privileges and can only delete images that they create.
- GH_IMG_TESTABLE: A role that enables users to add a working copy only when an image is in the `TESTABLE` state. Users assigned this role must also be assigned either the GH_WC_ADMIN role or the GH_WC_USER role to add a working copy.
- GH_IMG_RESTRICT: A role that enables users to add a working copy only when an image is in the `RESTRICTED` state. Users assigned this role must also be assigned either the GH_WC_ADMIN role or the GH_WC_USER role to add a working copy.
- GH_IMG_PUBLISH: Users assigned this role can promote an image to another state or retract an image from the `PUBLISHED` state to either the `TESTABLE` or `RESTRICTED` state.
- GH_IMG_VISIBILITY: Users assigned this role can modify access to promoted or published images using the `rhpctl allow | disallow image` commands.

Composite Built-In Roles

The composite built-in roles and their functions are:

- GH_SA: The Oracle Grid Infrastructure user on a Rapid Home Provisioning Server automatically inherits this role.

  The GH_SA role includes the following basic built-in roles: GH_ROLE_ADMIN, GH_SITE_ADMIN, GH_SERIES_ADMIN, GH_SERIES_CONTRIB, GH_WC_ADMIN, GH_IMG_ADMIN, GH_IMG_TESTABLE, GH_IMG_RESTRICT, GH_IMG_PUBLISH, and GH_IMG_VISIBILITY.
- GH_CA: The Oracle Grid Infrastructure user on a Rapid Home Provisioning Client automatically inherits this role.

  The GH_CA role includes the following basic built-in roles: GH_SERIES_ADMIN, GH_SERIES_CONTRIB, GH_WC_ADMIN, GH_IMG_ADMIN, GH_IMG_TESTABLE, GH_IMG_RESTRICT, GH_IMG_PUBLISH, and GH_IMG_VISIBILITY.
- GH_OPER: This role includes the following built-in roles: GH_WC_OPER, GH_SERIES_ADMIN, GH_IMG_TESTABLE, GH_IMG_RESTRICT, and GH_IMG_USER. Users assigned this role have limited access and cannot delete images.

Consider a gold image called `G1` that is available on the Rapid Home Provisioning Server with the GH_WC_USER user role. This role allows users read-only permission on `G1`.

Further consider that a user, `U1`, on a Rapid Home Provisioning Client, `Cl1`, has the GH_WC_USER role. If `U1` requests to provision an Oracle home based on the gold image `G1`, then `U1` can do so, because of the permissions granted by the GH_WC_USER role, which is also assigned to `G1`. If `U1` requests to delete `G1`, however, then that request would be denied because the GH_WC_USER role does not have the necessary permissions.

The Rapid Home Provisioning Server can associate user-role mappings to the Rapid Home Provisioning Client. After the Rapid Home Provisioning Server delegates user-role mappings, the Rapid Home Provisioning Client can then modify user-role mappings on the Rapid Home Provisioning Server for all users that belong to the Rapid Home Provisioning Client. This is implied by the fact that only the Rapid Home Provisioning Server qualifies user IDs from a Rapid Home Provisioning Client site with the client cluster name of that site. Thus, the Rapid Home Provisioning Client `CL1` will not be able to update user mappings of a user on `CL2`, where `CL2` is the cluster name of a different Rapid Home Provisioning Client.

Rapid Home Provisioning Images

By default, when you create a gold image using either `rhpctl import image` or `rhpctl add image`, the image is ready to provision working copies. However, under certain conditions, you may want to restrict access to images to enable someone to test or validate the image before making it available for general use.

Image State

You can set the state of an image to `TESTABLE` or `RESTRICTED` so that only users with the GH_IMG_TESTABLE or GH_IMG_RESTRICT roles can provision working copies from this image. Once the image has been tested or validated, you can change the state and make the image available for general use by running the `rhpctl promote image -image` *`image_name`* `-state PUBLISHED` command. The default image state is `PUBLISHED` when you add a new gold image, and if you do not specify the state in either the `rhpctl add image` or `rhpctl import image` commands.

Image Series

Using an image series is a convenient way to group different images into a logical sequence. Rapid Home Provisioning treats each image as an independent entity with respect to other images. No relationship is assumed between images, even if they follow some specific nomenclature. The image administrator names images in some logical manner that makes sense to its user community.

Use the `rhpctl add series` command to create an image series and associate one or more images to this series. The list of images in an image series is an ordered list. Use the `rhpctl insertimage series` and `rhpctl deleteimage series` to add and delete images in an image series. You can also change the order of images in a series using these commands.

Implementing Rapid Home Provisioning

After you install and configure Oracle Clusterware, you can configure and start Rapid Home Provisioning.

This section includes the following topics:

- Creating a Rapid Home Provisioning Server
- Adding Gold Images to the Rapid Home Provisioning Server
- Provisioning Software
- Creating a Rapid Home Provisioning Client

Creating a Rapid Home Provisioning Server

The Rapid Home Provisioning Server uses a repository that you create in an Oracle ACFS file system in which you store all the software homes that you want to make available to clients.

To create a Rapid Home Provisioning Server:

1. Use the Oracle ASM configuration assistant (ASMCA) to create an Oracle ASM disk group on the Rapid Home Provisioning Server to store software, as follows:

   ```
   $ Grid_home/bin/asmca
   ```

   Because this disk group is used to store software, Oracle recommends a minimum of 1 TB for this disk group.

   > **Note:** You must set Oracle ASM Dynamic Volume Manager (Oracle ADVM) compatibility settings for this disk group to `12.1`.

   > **See Also:** *Oracle Automatic Storage Management Administrator's Guide* for information about setting Oracle ASM disk group compatibility.

2. Provide a mount path that exists on all nodes of the cluster. The Rapid Home Provisioning Server uses this path to mount gold images.

   ```
   $ mkdir -p storage_path/images
   ```

3. As `root`, create the Rapid Home Provisioning Server resource, as follows:

   ```
   # Grid_home/bin/srvctl add rhpserver -storage storage_path
       -diskgroup disk_group_name
   ```

4. Start the Rapid Home Provisioning Server, as follows:

   ```
   $ Grid_home/bin/srvctl start rhpserver
   ```

   > **See Also:** "SRVCTL Command Reference" on page F-18 for more information about the SRVCTL commands used in this procedure

After you start the Rapid Home Provisioning Server, use the Rapid Home Provisioning Control (RHPCTL) utility to further manage Rapid Home Provisioning.

> **See Also:** Appendix F, "Rapid Home Provisioning and Server Control Command Reference" for more information about RHPCTL

Adding Gold Images to the Rapid Home Provisioning Server

The Rapid Home Provisioning Server stores and serves gold images of software homes. These images must be instantiated on the Rapid Home Provisioning Server.

Note: Images are read-only, and you cannot run programs from them. To create a usable software home from an image, you must create a working copy. You cannot directly use images as software homes. You can, however, use images to create working copies (software homes).

You can import software to the Rapid Home Provisioning Server using any one of the following methods:

- You can import an image from an installed home on the Rapid Home Provisioning Server using the following command:

  ```
  rhpctl import image -image image_name -path path_to_installed_home
    [-imagetype ORACLEDBSOFTWARE | SOFTWARE]
  ```

- You can import an image from an installed home on a Rapid Home Provisioning Client, using the following command run from the Rapid Home Provisioning Client:

  ```
  rhpctl import image -image image_name -path path_to_installed_home
  ```

- You can create an image from an existing working copy using the following command:

  ```
  rhpctl add image -image image_name -workingcopy working_copy_name
  ```

The preceding commands create an Oracle ACFS file system in the Rapid Home Provisioning root directory, similar to the following:

```
/u01/rhp/images/images/RDBMS_121020617524
```

See Also: Appendix F, "Rapid Home Provisioning and Server Control Command Reference" for more information about the preceding RHPCTL commands

Provisioning Software

After you import an image, you can provision software by adding a working copy either on the Rapid Home Provisioning Server or on the Rapid Home Provisioning Client. You can run the software provisioning command on either the Server or the Client.

1. To provision software on the Rapid Home Provisioning Server:

   ```
   rhpctl add workingcopy -workingcopy working_copy_name -image image_name
   ```

2. To create a working copy on the Rapid Home Provisioning Client:

   ```
   rhpctl add workingcopy -workingcopy working_copy_name -image image_name
      -storagetype storage_type -path path_to_software_home
   ```

3. To create a working copy on the Rapid Home Provisioning Client from the Rapid Home Provisioning Server:

   ```
   rhpctl add workingcopy -workingcopy working_copy_name -image image_name
      -path path_to_software_home -client client_cluster_name
   ```

Notes:

- The directory you specify in the `-path` parameter must be empty.
- You can re-run the provisioning command in case of an interruption or failure due to system or user errors. After you fix the reported errors, re-run the command and it will resume from the point of failure.

See Also: "Storage Space for Provisioned Software" on page 5-8 for more information

This section includes the following topics:

- Storage Space for Provisioned Software
- Provisioning for a Different User

Storage Space for Provisioned Software

You control storage space for provisioned software using the `-storagetype` and `-path` parameters of the `rhpctl add workingcopy` command. If you specify `-path`, then the application software will always be provisioned on the path you specify. If the file system of this path is shared among all of the nodes in the cluster, then the working copy will be created on this shared storage. If the file system is not shared, then the entire application software gets copied to the given path on every node in the cluster.

In addition to the `-path` parameter, if you specify `-storagetype`, then the type of storage you specify with this parameter is used. If you specify `-storagetype NFS`, then the path you provide is used as an NFS mount point. With NFS storage, the software resides on the Rapid Home Provisioning Server cluster, and is accessible as read/write on the Rapid Home Provisioning Client cluster over NFS. In this case, the same software is visible on all nodes of the remote cluster and is therefore shared across all nodes. You cannot specify NFS storage when provisioning software on the Rapid Home Provisioning Server cluster.

If you specify `-storagetype LOCAL`, then working copies are stored in user-provided file systems, and not in storage provided by Rapid Home Provisioning. If you did not specify the `-path` parameter, then a default Oracle home path, such as `oracle_base_path/product/db_version/workingcopy_name`, based on the value of the `-oraclebase` option is used to provision the database software. This path could be shared or non-shared.

If you do not specify the `-storagetype` and `-path` parameters, then the Rapid Home Provisioning Server uses the default storage type, `RHP_MANAGED`, and the Rapid Home Provisioning Server creates an Oracle ACFS shared file system. If you do not specify the `-path` parameter, then Rapid Home Provisioning creates an Oracle ACFS shared file system on the target cluster and makes the software available to the user from this file system. This can happen on the Rapid Home Provisioning Client cluster only if you configured the Rapid Home Provisioning Client with an Oracle ASM disk group.

Provisioning for a Different User

If you want a different user to provision software, then use the `-user` and `-client` parameters of the `rhpctl add workingcopy` command. If the provisioning is for the cluster from where you run the command, then you do not need to specify the `-client` parameter. When the provisioning is completed, all files and directories of the provisioned software are owned by the user you specified. Permissions on files on the remotely provisioned software are the same as the permissions that existed on the gold image from where you provisioned the application software.

Creating a Rapid Home Provisioning Client

Users operate on a Rapid Home Provisioning Client to perform tasks such as requesting deployment of Oracle homes and querying gold images.

To create a Rapid Home Provisioning Client:

1. If there is no highly available VIP (HAVIP) on the Rapid Home Provisioning Server, then, as the `root` user, create an HAVIP, as follows:

   ```
   # srvctl add havip -id id -address {host_name | ip_address}
   ```

   You can specify either a host name or IPv4 or IPv6 IP address. The IP address that you specify for HAVIP or the address that is resolved from the specified host name *must not* be in use when you run this command.

   Note: The highly available VIP must be in the same subnet as the default network configured in the Rapid Home Provisioning Server cluster. You can obtain the subnet by running the following command:

   ```
   srvctl config network -netnum network_number
   ```

   See Also: *Oracle Real Application Clusters Administration and Deployment Guide* for more information about these SRVCTL commands

2. On the Rapid Home Provisioning Server as the Grid home owner, create the client data file, as follows:

   ```
   $ rhpctl add client -client client_cluster_name -toclientdata path
   ```

   RHPCTL creates the client data file in the directory path you specify after the `-toclientdata` flag. The name of the client data file is *`client_cluster_name`*`.xml`.

   Note: The *`client_cluster_name`* must be unique and it must match the cluster name of the client cluster where you run step 4.

   See Also: Appendix F, "Rapid Home Provisioning and Server Control Command Reference" for more information about the `rhpctl add client` command

3. Copy the client data file that you created in the previous step to a directory on the client cluster that has read/write permissions to the Grid home owner on the Rapid Home Provisioning Client.
4. Create the Rapid Home Provisioning Client by running the following command as `root` on the client cluster:

   ```
   # srvctl add rhpclient -clientdata path_to_client_data
      [-diskgroup disk_group_name -storage base_path]
   ```

   If you want to provision working copies to Oracle ACFS storage on this cluster, and you have already created a disk group for this purpose, then specify this disk group in the preceding command. In this case, also specify a storage path which will be used as a base path for all mount points when creating Oracle ACFS file systems for storing working copies.

> **Note:** Once you configure a disk group on a Rapid Home Provisioning Client, you cannot remove it from or change it in the Rapid Home Provisioning Client configuration. The only way you can do either (change or remove) is to completely remove the Rapid Home Provisioning Client using the `srvctl remove client` command, and then add it back with a different disk group, if necessary. Before you remove a Rapid Home Provisioning Client, ensure that you remove all registered users from this cluster and all working copies provisioned on this cluster.

5. Start the Rapid Home Provisioning Client, as follows:

```
$ srvctl start rhpclient
```

6. Check the status of the Rapid Home Provisioning Client, as follows:

```
$ srvctl status rhpclient
```

Managing Rapid Home Provisioning Clients

Rapid Home Provisioning Client management tasks include:

- Enabling and Disabling Rapid Home Provisioning Clients
- Creating Users and Assigning Roles for Rapid Home Provisioning Client Cluster Users
- Managing the Rapid Home Provisioning Client Password

Enabling and Disabling Rapid Home Provisioning Clients

Rapid Home Provisioning Clients communicate with the Rapid Home Provisioning Server for all actions. You cannot run any RHPCTL commands without a connection to a Rapid Home Provisioning Server.

From the Rapid Home Provisioning Server, you can enable or disable Rapid Home Provisioning Clients by running the following command from the Rapid Home Provisioning Server cluster:

```
$ rhpctl modify client -client client_name -enabled TRUE | FALSE
```

To enable a Rapid Home Provisioning Client, specify `-enabled TRUE`. Conversely, specify `-enabled FALSE` to disable the client. When you disable a Rapid Home Provisioning Client cluster, all RHPCTL commands from that client cluster will be rejected by the Rapid Home Provisioning Server, unless and until you re-enable the client.

> **Note:** Disabling a Rapid Home Provisioning Client cluster does not disable any existing working copies on the client cluster. The working copies will continue to function and any databases in those working copies will continue to run, even if their Oracle home storage comes from the Rapid Home Provisioning Server cluster over NFS.

Creating Users and Assigning Roles for Rapid Home Provisioning Client Cluster Users

Use the `-maproles` parameter with the `rhpctl add client` command to create users and assign roles to Rapid Home Provisioning Client users. You can associate multiple users with roles, or you can assign a single user multiple roles with this command.

Managing the Rapid Home Provisioning Client Password

The Rapid Home Provisioning Client uses a password stored internally to authenticate itself with the Rapid Home Provisioning Server. You cannot query this password, however, if for some reason, you are required to reset this password, then you can do so, as follows, on the Rapid Home Provisioning Server cluster:

1. Run the following command on the Rapid Home Provisioning Server cluster to generate a new password and store it in the client credential:

   ```
   $ rhpctl modify client -client client_name -password
   ```

2. Run the following command on the Rapid Home Provisioning Server cluster to generate a credential file:

   ```
   $ rhpctl export client -client client_name -clientdata file_path
   ```

   For example, to generate a credential file for a Rapid Home Provisioning Client named `mjk9394`:

   ```
   $ rhpctl export client -client mjk9394 -clientdata /tmp/mjk9394.xml
   ```

3. Continuing with the preceding example, transport the generated credential file securely to the Rapid Home Provisioning Client cluster and then run the following command on any node in the Rapid Home Provisioning Client cluster:

   ```
   $ srvctl modify rhpclient -clientdata path_to_mjk9394.xml
   ```

4. Restart the Rapid Home Provisioning Client daemon by running the following commands on the Rapid Home Provisioning Client cluster:

   ```
   $ srvctl stop rhpclient
   $ srvctl start rhpclient
   ```

6

Managing Oracle Cluster Registry and Voting Files

Oracle Clusterware includes two important components that manage configuration and node membership: Oracle Cluster Registry (OCR), which also includes the local component Oracle Local Registry (OLR), and voting files.

- OCR manages Oracle Clusterware and Oracle RAC database configuration information
- OLR resides on every node in the cluster and manages Oracle Clusterware configuration information for each particular node
- Voting files manage information about node membership. Each voting file must be accessible by all nodes in the cluster for nodes to be members of the cluster

You can store OCR and voting files on Oracle Automatic Storage Management (Oracle ASM), or a shared file system.

> **Note:** Oracle Clusterware 12*c* does not support the use of raw or block devices. To upgrade to Oracle Clusterware 12*c* from a previous Oracle Clusterware release on which you were using raw or block devices, you must migrate OCR and voting files to Oracle ASM or a shared file system before you upgrade.

Oracle recommends that you configure multiple voting files during Oracle Clusterware installation to improve availability. If you choose to put the voting files into an Oracle ASM disk group, then Oracle ASM ensures the configuration of multiple voting files if you use a normal or high redundancy disk group. If you choose to store the voting files on a shared file system, then select the option to configure multiple voting files, in which case you will have to specify three different file systems based on different disks.

If necessary, you can dynamically add or replace voting files after you complete the Oracle Clusterware installation process without stopping the cluster.

This chapter includes the following topics:

Managing Oracle Cluster Registry and Oracle Local Registry

This section describes how to manage OCR and the Oracle Local Registry (OLR) with the following utilities: OCRCONFIG, OCRDUMP, and OCRCHECK.

OCR contains information about all Oracle resources in the cluster.

OLR is a registry similar to OCR located on each node in a cluster, but contains information specific to each node. It contains manageability information about Oracle Clusterware, including dependencies between various services. Oracle High Availability Services uses this information. OLR is located on local storage on each node in a cluster. Its default location is in the path `Grid_home/cdata/host_name.olr`, where `Grid_home` is the Oracle Grid Infrastructure home, and `host_name` is the host name of the node.

This section describes how to administer OCR in the following topics:

- Migrating Oracle Cluster Registry to Oracle Automatic Storage Management
- Adding, Replacing, Repairing, and Removing Oracle Cluster Registry Locations
- Backing Up Oracle Cluster Registry
- Restoring Oracle Cluster Registry
- Diagnosing Oracle Cluster Registry Problems
- Administering Oracle Cluster Registry with Export and Import Commands
- Oracle Local Registry
- Upgrading and Downgrading the Oracle Cluster Registry Configuration

> **See Also:** "About OCRCONFIG" on page I-1 for information about the OCRCONFIG utility, and "Troubleshooting Oracle Cluster Registry" on page I-10 for information about the OCRDUMP and OCRCHECK utilities

Migrating Oracle Cluster Registry to Oracle Automatic Storage Management

To improve Oracle Clusterware storage manageability, OCR is configured, by default, to use Oracle ASM in Oracle Database 12*c*. With the Oracle Clusterware storage residing in an Oracle ASM disk group, you can manage both database and clusterware storage using Oracle Enterprise Manager.

However, if you upgrade from a previous version of Oracle Clusterware, you can migrate OCR to reside on Oracle ASM, and take advantage of the improvements in managing Oracle Clusterware storage.

> **Note:** If you upgrade from a previous version of Oracle Clusterware to Oracle Clusterware 12*c* and you want to store OCR in an Oracle ASM disk group, then you must set the **ASM Compatibility** compatibility attribute to `11.2.0.2`, or later.

> **See Also:** *Oracle Automatic Storage Management Administrator's Guide* for information about setting Oracle ASM compatibility attributes

To migrate OCR to Oracle ASM using OCRCONFIG:

1. Ensure the upgrade to Oracle Clusterware 12*c* is complete. Run the following command to verify the current running version:

```
$ crsctl query crs activeversion
```

2. Use the Oracle ASM Configuration Assistant (ASMCA) to configure and start Oracle ASM on all nodes in the cluster.

   See Also: *Oracle Automatic Storage Management Administrator's Guide* for more information about using ASMCA

3. Use ASMCA to create an Oracle ASM disk group that is at least the same size of the existing OCR and has at least normal redundancy.

   Notes:

   - If OCR is stored in an Oracle ASM disk group with external redundancy, then Oracle recommends that you add another OCR location to another disk group to avoid the loss of OCR, if a disk fails in the disk group.

     Oracle does not support storing OCR on different storage types simultaneously, such as storing OCR on both Oracle ASM and a shared file system, except during a migration.

   - If an Oracle ASM instance fails on any node, then OCR becomes unavailable on that particular node.

     If the CRSD process running on the node affected by the Oracle ASM instance failure is the **OCR writer**, the majority of the OCR locations are stored in Oracle ASM, and you attempt I/O on OCR during the time the Oracle ASM instance is down on this node, then CRSD stops and becomes inoperable. Cluster management is now affected on this particular node.

     Under no circumstances will the failure of one Oracle ASM instance on one node affect the whole cluster.

   - Ensure that Oracle ASM disk groups that you create are mounted on all of the nodes in the cluster.

   See Also: *Oracle Grid Infrastructure Installation Guide* for more detailed sizing information

4. To add OCR to an Oracle ASM disk group, ensure that the **Oracle Clusterware stack** is running and run the following command as `root`:

   ```
   # ocrconfig -add +new_disk_group
   ```

   You can run this command more than once if you add multiple OCR locations. You can have up to five OCR locations. However, each successive run must point to a different disk group.

5. To remove storage configurations no longer in use, run the following command as `root`:

   ```
   # ocrconfig -delete old_storage_location
   ```

   Run this command for every configured OCR.

The following example shows how to migrate two OCRs to Oracle ASM using OCRCONFIG.

```
# ocrconfig -add +new_disk_group
# ocrconfig -delete /ocrdata/ocr_1
# ocrconfig -delete /ocrdata/ocr_2
```

Note:

- OCR inherits the redundancy of the disk group. If you want high redundancy for OCR, you must configure the disk group with high redundancy when you create it.
- Oracle recommends that you put the SPFILE for Oracle ASM in this newly-created OCR location.

See Also: *Oracle Automatic Storage Management Administrator's Guide* for information about setting the location of the Oracle ASM SPFILE

Migrating Oracle Cluster Registry from Oracle ASM to Other Types of Storage

To migrate OCR from Oracle ASM to another storage type:

1. Ensure the upgrade to Oracle Clusterware 12*c* is complete. Run the following command to verify the current running version:

   ```
   $ crsctl query crs activeversion
   ```

2. Create a file in a shared or cluster file system with the following permissions: `root`, `oinstall`, `640`.

   Note: Create at least two mirrors of the primary storage location to eliminate a single point of failure for OCR. OCR supports up to five locations.

3. Ensure there is at least 500 MB of space on the mount partition.
4. Ensure that the file you created is visible from all nodes in the cluster.
5. To add the file as an OCR location, ensure that the Oracle Clusterware stack is running and run the following command as `root`:

   ```
   # ocrconfig -add file_location
   ```

   You can run this command more than once if you add more than one OCR location. Each successive run of this command must point to a different file location.

6. Ensure that the OCR backup location is not configured on an Oracle ASM disk group.

   To migrate the OCR backup location to a file system, run the following command as `root`:

   ```
   # ocrconfig -backuploc file_location
   ```

7. To remove storage configurations no longer in use, run the following command as `root`:

   ```
   # ocrconfig -delete +asm_disk_group
   ```

   You can run this command more than once if there is more than one OCR location configured.

The following example shows how to migrate OCR from Oracle ASM to NAS using OCRCONFIG.

```
# ocrconfig -add /nas/ocr
```

For OCRs not stored on Oracle ASM, Oracle recommends that you mirror OCR on different devices.

Adding, Replacing, Repairing, and Removing Oracle Cluster Registry Locations

The Oracle installation process for Oracle Clusterware gives you the option of automatically mirroring OCR. You can manually put the mirrored OCRs on a shared Network File System (NFS), or on any cluster file system that is certified by Oracle. Alternatively, you can place OCR on Oracle ASM and allow it to create mirrors automatically, depending on the redundancy option you select.

This section includes the following topics:

- Adding an Oracle Cluster Registry Location
- Removing an Oracle Cluster Registry Location
- Replacing an Oracle Cluster Registry Location
- Repairing an Oracle Cluster Registry Configuration on a Local Node
- Overriding the Oracle Cluster Registry Data Loss Protection Mechanism

You can manually mirror OCR, as described in the "Adding an Oracle Cluster Registry Location" on page 6-6 section, if you:

- Upgraded to Oracle Clusterware 12*c* but did not choose to mirror OCR during the upgrade
- Created only one OCR location during the Oracle Clusterware installation

Oracle recommends that you configure:

- At least three OCR locations, if OCR is configured on non-mirrored or non-redundant storage. Oracle strongly recommends that you mirror OCR if the underlying storage is not RAID. Mirroring can help prevent OCR from becoming a single point of failure.
- At least two OCR locations if OCR is configured on an Oracle ASM disk group. You should configure OCR in two independent disk groups. Typically this is the work area and the recovery area.
- At least two OCR locations if OCR is configured on mirrored hardware or third-party mirrored volumes.

Notes:

- If the original OCR location does not exist, then you must create an empty (0 byte) OCR location with appropriate permissions before you run the `ocrconfig -add` or `ocrconfig -replace` commands.
- Ensure that the OCR devices that you specify in the OCR configuration exist and that these OCR devices are valid.
- Ensure that the Oracle ASM disk group that you specify exists and is mounted.
- The new OCR file, device, or disk group must be accessible from all of the active nodes in the cluster.

See Also:

- *Oracle Grid Infrastructure Installation Guide* for information about creating OCRs
- *Oracle Automatic Storage Management Administrator's Guide* for more information about Oracle ASM disk group management

In addition to mirroring OCR locations, you can also:

- Replace an OCR location if there is a misconfiguration or other type of OCR error, as described in the "Replacing an Oracle Cluster Registry Location" on page 6-7 section.
- Repair an OCR location if Oracle Database displays an OCR failure alert in Oracle Enterprise Manager or in the Oracle Clusterware alert log file, as described in the "Repairing an Oracle Cluster Registry Configuration on a Local Node" on page 6-8 section.
- Remove an OCR location if, for example, your system experiences a performance degradation due to OCR processing or if you transfer your OCR to RAID storage devices and choose to no longer use multiple OCR locations, as described in the "Removing an Oracle Cluster Registry Location" on page 6-7 section.

Note: The operations in this section affect OCR clusterwide: they change the OCR configuration information in the `ocr.loc` file on Linux and UNIX systems and the Registry keys on Windows systems. However, the `ocrconfig` command cannot modify OCR configuration information for nodes that are shut down or for nodes on which Oracle Clusterware is not running.

Adding an Oracle Cluster Registry Location

Use the procedure in this section to add an OCR location. Oracle Clusterware can manage up to five redundant OCR locations.

Note: If OCR resides on a cluster file system file or a network file system, create an empty (0 byte) OCR location file before performing the procedures in this section.

As the root user, run the following command to add an OCR location to either Oracle ASM or other storage device:

```
# ocrconfig -add +asm_disk_group | file_name
```

> **Note:** On Linux and UNIX systems, you must be root to run ocrconfig commands. On Windows systems, the user must be a member of the Administrator's group.

Removing an Oracle Cluster Registry Location

To remove an OCR location or a failed OCR location, at least one other OCR must be online. You can remove an OCR location to reduce OCR-related overhead or to stop mirroring your OCR because you moved OCR to redundant storage such as RAID.

Perform the following procedure as the root user to remove an OCR location from your Oracle Clusterware environment:

1. Ensure that *at least one* OCR location other than the OCR location that you are removing is online.

   > **Caution:** Do *not* perform this OCR removal procedure unless there is at least one other active OCR location online.

2. Run the following command on any node in the cluster to remove an OCR location from either Oracle ASM or other location:

   ```
   # ocrconfig -delete +ASM_disk_group | file_name
   ```

   The *file_name* variable can be a device name or a file name. This command updates the OCR configuration on all of the nodes on which Oracle Clusterware is running.

   > **Note:** The only configured Oracle ASM OCR location, which is on a different Oracle ASM disk group than the Oracle ASM OCR backup location, cannot be removed.

Replacing an Oracle Cluster Registry Location

If you must change an existing OCR location, or change a failed OCR location to a working location, then you can use the following procedure, if all remaining OCR locations remain online. The ocrconfig -replace command requires that at least two OCR locations are configured.

To change an Oracle Cluster Registry location:

Complete the following procedure:

1. Use the OCRCHECK utility to verify that a copy of OCR other than the one you are going to replace is *online*, using the following command:

   ```
   $ ocrcheck
   ```

   OCRCHECK displays all OCR locations that are registered and whether they are available (online). If an OCR location suddenly becomes unavailable, then it might take a short period for Oracle Clusterware to show the change in status.

Note: The OCR location that you are *replacing* can be either online or offline.

2. Use the following command to verify that Oracle Clusterware is running on the node on which you are going to perform the replace operation:

   ```
   $ crsctl check crs
   ```

3. Run the following command as `root` to replace the current OCR location using either *`destination_file`* or *`+ASM_disk_group`* to indicate the current and target OCR locations:

   ```
   # ocrconfig -replace current_OCR_location -replacement new_OCR_location
   ```

   The preceding command fails if you have less than two configured OCR locations that are online.

   If you have only one OCR location configured and online, then you must first add a new location and then delete the failed location, as follows:

   ```
   # ocrconfig -add new_OCR_location
   # ocrconfig -delete current_OCR_location
   ```

Note: If your cluster configuration changes while the node on which OCR resides is stopped, and the Oracle Clusterware stack is running on the other nodes, then OCR detects configuration changes and self-corrects the configuration by changing the contents of the `ocr.loc` file.

See Also: *Oracle Automatic Storage Management Administrator's Guide* for more information about migrating storage

Repairing an Oracle Cluster Registry Configuration on a Local Node

It may be necessary to repair OCR if your cluster configuration changes while that node is stopped and this node is the only member in the cluster. Repairing an OCR involves either adding, deleting, or replacing an OCR location. For example, if any node that is part of your current Oracle RAC cluster is shut down, then you must update the OCR configuration on the stopped node to let that node rejoin the cluster after the node is restarted. Use the following command syntax as `root` on the restarted node where you use either a *`destination_file`* or *`+ASM_disk_group`* to indicate the current and target OCR locations:

```
ocrconfig -repair -replace current_OCR_location -replacement target_OCR_location
```

This operation only changes OCR on the node on which you run this command. For example, if the OCR location is `/dev/sde1`, then use the command syntax `ocrconfig -repair -add /dev/sde1` on this node to repair OCR on that node.

Notes:

- You cannot repair the OCR configuration on a node on which the Oracle Cluster Ready Services daemon is running.
- When you repair OCR on a stopped node using `ocrconfig -repair`, you must provide the same OCR file name (which should be case-sensitive) as the OCR file names on other nodes.
- If you run the `ocrconfig -add | -repair | -replace` command, then the device, file, or Oracle ASM disk group that you are adding must be accessible. This means that a device must exist. You must create an empty (0 byte) OCR location, or the Oracle ASM disk group must exist and be mounted.

See Also:

- "OCRCONFIG Command Reference" on page I-1 for more information about OCRCONFIG commands
- *Oracle Automatic Storage Management Administrator's Guide* for more information about Oracle ASM disk group management

Overriding the Oracle Cluster Registry Data Loss Protection Mechanism

OCR has a mechanism that prevents data loss due to accidental overwrites. If you configure a mirrored OCR and if Oracle Clusterware cannot access the mirrored OCR locations and also cannot verify that the available OCR location contains the most recent configuration, then Oracle Clusterware prevents further modification to the available OCR location. In addition, the process prevents overwriting by prohibiting Oracle Clusterware from starting on the node on which only one OCR is available. In such cases, Oracle Database displays an alert message in either Oracle Enterprise Manager, the Oracle Clusterware alert log files, or both. If this problem is local to only one node, you can use other nodes to start your cluster database.

However, if you are unable to start any cluster node in your environment and if you can neither repair OCR nor restore access to all OCR locations, then you can override the protection mechanism. The procedure described in the following list enables you to start the cluster using the available OCR location. However, overriding the protection mechanism can result in the loss of data that was not available when the previous known good state was created.

Caution: Overriding OCR using the following procedure can result in the loss of OCR updates that were made between the time of the last known good OCR update made to the currently accessible OCR and the time at which you performed the overwrite. In other words, running the `ocrconfig -overwrite` command can result in data loss if the OCR location that you are using to perform the overwrite does not contain the latest configuration updates for your cluster environment.

Perform the following procedure to overwrite OCR if a node cannot start *and* if the alert log contains CLSD-1009 and CLSD-1011 messages.

1. Attempt to resolve the cause of the CLSD-1009 and CLSD-1011 messages.

Compare the node's OCR configuration (`ocr.loc` on Linux and UNIX systems and the Registry on Windows systems) with other nodes on which Oracle Clusterware is running.

- If the configurations do not match, run `ocrconfig -repair`.
- If the configurations match, ensure that the node can access all of the configured OCRs by running an `ls` command on Linux and UNIX systems. On Windows, use a `dir` command if the OCR location is a file and run `GuiOracleObjectManager.exe` to verify that the part of the cluster with the name exists.

2. Ensure that the most recent OCR contains the latest OCR updates.

   Look at output from the `ocrdump` command and determine whether it has your latest updates.

3. If you cannot resolve the problem that caused the CLSD message, then run the command `ocrconfig -overwrite` to start the node.

Backing Up Oracle Cluster Registry

This section describes how to back up OCR content and use it for recovery. The first method uses automatically generated OCR copies and the second method enables you to issue a backup command manually:

- **Automatic backups**: Oracle Clusterware automatically creates OCR backups every four hours. At any one time, Oracle Database always retains the last three backup copies of OCR. The CRSD process that creates the backups also creates and retains an OCR backup for each *full day* and *after each week*. You cannot customize the backup frequencies or the number of files that Oracle Database retains.
- **Manual backups**: Run the `ocrconfig -manualbackup` command on a node where the Oracle Clusterware stack is up and running to force Oracle Clusterware to perform a backup of OCR at any time, rather than wait for the automatic backup. You must run the command as a user with administrative privileges. The `-manualbackup` option is especially useful when you want to obtain a binary backup on demand, such as before you make changes to OCR. The OLR only supports manual backups.

When the clusterware stack is down on all nodes in the cluster, the backups that are listed by the `ocrconfig -showbackup` command may differ from node to node.

> **Note:** After you install or upgrade Oracle Clusterware on a node, or add a node to the cluster, when the `root.sh` script finishes, it backs up OLR.

This section includes the following topics:

- Listing Backup Files
- Changing Backup Location

Listing Backup Files

Run the following command to list the backup files:

```
ocrconfig -showbackup
```

The `ocrconfig -showbackup` command displays the backup location, timestamp, and the originating node name of the backup files that Oracle Clusterware creates. By default, the `-showbackup` option displays information for both automatic and manual backups but you can include the `auto` or `manual` flag to display only the automatic backup information or only the manual backup information, respectively.

Run the following command to inspect the contents and verify the integrity of the backup file:

```
ocrdump -backupfile backup_file_name
```

You can use any backup software to copy the automatically generated backup files at least once daily to a different device from where the primary OCR resides.

The default location for generating backups on Linux or UNIX systems is *`Grid_home`*`/cdata/`*`cluster_name`*, where *`cluster_name`* is the name of your cluster. The Windows default location for generating backups uses the same path structure. Because the default backup is on a local file system, Oracle recommends that you include the backup file created with the OCRCONFIG utility as part of your operating system backup using standard operating system or third-party tools.

Changing Backup Location

Run the following command to change the location where OCR creates backups:

```
ocrconfig -backuploc file_name
```

The *`file_name`* variable in the preceding command can be a full directory path name that is accessible by all nodes, or it can be an Oracle ASM disk group that is mounted on all nodes. You must migrate OCR to Oracle ASM before changing the OCR backup location to an Oracle ASM disk group. You can change the OCR backup location to an Oracle ASM disk group only if there is at least one Oracle ASM OCR location in a separate disk group.

For example, to specify an OCR backup location in a directory:

```
# ocrconfig -backuploc Grid_home/cdata/cluster3
```

To specify an OCR backup location in an Oracle ASM disk group:

```
# ocrconfig -backuploc +bkupdg
```

Note: On Linux and UNIX systems, you must be `root` user to run most but not all of the `ocrconfig` command options. On Windows systems, the user must be a member of the Administrator's group.

See Also:

- "Administering Oracle Cluster Registry with Export and Import Commands" on page 6-17 to use manually created OCR export files to copy OCR content and use it for recovery
- "OCRCONFIG Command Reference" on page I-1 for more information about OCRCONFIG commands

Restoring Oracle Cluster Registry

If a resource fails, then before attempting to restore OCR, restart the resource. As a definitive verification that OCR failed, run `ocrcheck` and if the command returns a failure message, then both the primary OCR and the OCR mirror have failed. Attempt to correct the problem using the OCR restoration procedure for your platform.

Notes:

- You *cannot* restore your configuration from an OCR backup file using the `-import` option, which is explained in "Administering Oracle Cluster Registry with Export and Import Commands" on page 6-17. You *must instead* use the `-restore` option, as described in the following sections.
- If you store OCR on an Oracle ASM disk group and the disk group is not available, then you must recover and mount the Oracle ASM disk group.

See Also: *Oracle Automatic Storage Management Administrator's Guide* for more information about managing Oracle ASM disk groups

- Restoring the Oracle Cluster Registry on Linux or UNIX Systems
- Restoring the Oracle Cluster Registry on Windows Systems
- Restoring the Oracle Cluster Registry in an Oracle Restart Environment

Restoring the Oracle Cluster Registry on Linux or UNIX Systems

If you are storing OCR on an Oracle ASM disk group, and that disk group is corrupt, then you must restore the Oracle ASM disk group using Oracle ASM utilities, and then mount the disk group again before recovering OCR. Recover OCR by running the `ocrconfig -restore` command, as instructed in the following procedure.

Note: If the original OCR location does not exist, then you must create an empty (0 byte) OCR location with the same name as the original OCR location before you run the `ocrconfig -restore` command.

See Also: *Oracle Automatic Storage Management Administrator's Guide* for information about how to restore Oracle ASM disk groups

Use the following procedure to restore OCR on Linux or UNIX systems:

1. List the nodes in your cluster by running the following command on one node:

   ```
   $ olsnodes
   ```

2. Depending on whether OCR is located in an Oracle ASM disk group or on network attached storage (NAS), stop Oracle Clusterware by running one of the following commands as `root` on all of the nodes:

   If OCR is located in an Oracle ASM disk group, then stop the Oracle Clusterware daemon:

   ```
   # crsctl stop crs
   ```

   If the preceding command returns any error due to OCR corruption, then force stop Oracle Clusterware by running the following command as `root` on all of the nodes:

```
# crsctl stop crs -f
```

3. If you are restoring OCR to a cluster file system or network file system, then run the following command as `root` to restore OCR with an OCR backup that you can identify in "Listing Backup Files" on page 6-10:

   ```
   # ocrconfig -restore file_name
   ```

   > **Note:** If the OCR backup file is located in an Oracle ASM disk group, then ensure that the disk group exists and is mounted.

   After you complete this step, proceed to step 11.

   If you are *not* restoring OCR to a cluster file system or network file system, then proceed to the next step.

4. Start the Oracle Clusterware stack on one node in exclusive mode by running the following command as `root`:

   ```
   # crsctl start crs -excl -nocrs
   ```

   The `-nocrs` option ensures that the CRSD process and OCR do not start with the rest of the Oracle Clusterware stack.

   Ignore any errors that display.

5. Check whether CRSD is running by running the following command:

   ```
   $ crsctl status resource ora.crsd -init
   ```

   If CRSD is running, then stop it by running the following command as `root`:

   ```
   # crsctl stop resource ora.crsd -init
   ```

   > **Caution:** *Do not* use the `-init` flag with any other command unless directed to do so by My Oracle Support.

6. To restore OCR to an Oracle ASM disk group, you must first create a disk group using SQL*Plus that has the same name as the disk group you want to restore and mount it on the local node.

   If you cannot mount the disk group locally, then run the following SQL*Plus command:

   ```
   SQL> drop diskgroup disk_group_name force including contents;
   ```

7. Restore OCR with an OCR backup that you can identify in "Listing Backup Files" on page 6-10 by running the following command as `root`:

   ```
   # ocrconfig -restore file_name
   ```

Notes:

- If the original OCR location does not exist, then you must create an empty (0 byte) OCR location before you run the `ocrconfig -restore` command.
- Ensure that the OCR devices that you specify in the OCR configuration exist and that these OCR devices are valid.
- If you configured OCR in an Oracle ASM disk group, then ensure that the Oracle ASM disk group exists and is mounted.
- If the OCR backup file is located in an Oracle ASM disk group, then ensure that the disk group exists and is mounted.

See Also:

- *Oracle Grid Infrastructure Installation Guide* for information about creating OCRs
- *Oracle Automatic Storage Management Administrator's Guide* for more information about Oracle ASM disk group management

8. Verify the integrity of OCR:

```
# ocrcheck
```

9. Stop Oracle Clusterware on the node where it is running in exclusive mode:

```
# crsctl stop crs -f
```

10. Run the `ocrconfig -repair -replace` command as `root` on all the nodes in the cluster where you *did not* the `ocrconfig -restore` command. For example, if you ran the `ocrconfig -restore` command on node 1 of a four-node cluster, then you must run the `ocrconfig -repair -replace` command on nodes 2, 3, and 4.
11. Begin to start Oracle Clusterware by running the following command as `root` on all of the nodes:

```
# crsctl start crs
```

12. Verify OCR integrity of all of the cluster nodes that are configured as part of your cluster by running the following CVU command:

```
$ cluvfy comp ocr -n all -verbose
```

See Also: Appendix A, "Cluster Verification Utility Reference" for more information about enabling and using CVU

Restoring the Oracle Cluster Registry on Windows Systems

If you are storing OCR on an Oracle ASM disk group, and that disk group is corrupt, then you must restore the Oracle ASM disk group using Oracle ASM utilities, and then mount the disk group again before recovering OCR. Recover OCR by running the `ocrconfig -restore` command.

Note: If the original OCR location does not exist, then you must create an empty (0 byte) OCR location with the same name as the original OCR location before you run the `ocrconfig -restore` command.

See Also: *Oracle Automatic Storage Management Administrator's Guide* for information about how to restore Oracle ASM disk groups

Use the following procedure to restore OCR on Windows systems:

1. List the nodes in your cluster by running the following command on one node:

   ```
   C:\>olsnodes
   ```

2. Stop Oracle Clusterware by running the following command as a member of the Administrators group on all of the nodes:

   ```
   C:\>crsctl stop crs
   ```

   If the preceding command returns any error due to OCR corruption, stop Oracle Clusterware by running the following command as a member of the Administrators group on all of the nodes:

   ```
   C:\>crsctl stop crs -f
   ```

3. Start the Oracle Clusterware stack on one node in exclusive mode by running the following command as a member of the Administrators group:

   ```
   C:\>crsctl start crs -excl -nocrs
   ```

   The `-nocrs` option ensures that the CRSD process and OCR do not start with the rest of the Oracle Clusterware stack.

   Ignore any errors that display.

4. Restore OCR with the OCR backup file that you identified in "Listing Backup Files" on page 6-10 by running the following command as a member of the Administrators group:

   ```
   C:\>ocrconfig -restore file_name
   ```

   Make sure that the OCR devices that you specify in the OCR configuration exist and that these OCR devices are valid.

Notes:

- Ensure that the OCR devices that you specify in the OCR configuration exist and that these OCR devices are valid.
- Ensure that the Oracle ASM disk group you specify exists and is mounted.

See Also:

- *Oracle Grid Infrastructure Installation Guide* for information about creating OCRs
- *Oracle Automatic Storage Management Administrator's Guide* for more information about Oracle ASM disk group management

5. Verify the integrity of OCR:

   ```
   C:\>ocrcheck
   ```

6. Stop Oracle Clusterware on the node where it is running in exclusive mode:

   ```
   C:\>crsctl stop crs -f
   ```

7. Begin to start Oracle Clusterware by running the following command as a member of the Administrators group on all of the nodes:

   ```
   C:\>crsctl start crs
   ```

8. Run the following Cluster Verification Utility (CVU) command to verify OCR integrity of all of the nodes in your cluster database:

   ```
   C:\>cluvfy comp ocr -n all -verbose
   ```

   See Also: Appendix A, "Cluster Verification Utility Reference" for more information about enabling and using CVU

Restoring the Oracle Cluster Registry in an Oracle Restart Environment

Notes:

- OCR is present for backward compatibility.
- Once an OCR location is created, it does not get updated in the Oracle Restart environment.
- If the Oracle Restart home has been backed up, and if there is a failure, then restoring the Oracle Restart home restores OCR.

Use the following procedure to restore OCR in an Oracle Restart environment:

1. Stop Oracle High Availability Services by running the following command as `root` on all of the nodes:

   ```
   # crsctl stop has [-f]
   ```

2. Run the `ocrcheck -config` command to determine the OCR location and then create an empty (0 byte) OCR location with appropriate permissions in that location.

3. Restore OCR by running the following command as `root`:

   ```
   # crsctl pin css -n host_name
   ```

   Notes: Ensure that the OCR devices that you specify in the OCR configuration exist and that these OCR devices are valid.

   See Also: *Oracle Grid Infrastructure Installation Guide* for information about creating OCRs

4. Run the `ocrcheck` command to verify the integrity of OCR:

5. Start Oracle High Availability Services by running the following command on all of the nodes:

```
$ crsctl start has
```

Diagnosing Oracle Cluster Registry Problems

You can use the OCRDUMP and OCRCHECK utilities to diagnose OCR problems.

See Also:

- "OCRDUMP Utility Syntax and Options" on page I-13 for more information about the OCRDUMP utility
- "Using the OCRCHECK Utility" on page I-10 for more information about the OCRCHECK utility

Administering Oracle Cluster Registry with Export and Import Commands

In addition to using the automatically created OCR backup files, you should also export OCR contents before and after making significant configuration changes, such as adding or deleting nodes from your environment, modifying Oracle Clusterware resources, and upgrading, downgrading or creating a database. Do this by using the `ocrconfig -export` command, which exports OCR content to a file format.

Caution: Note the following restrictions for restoring OCR:

- The file format generated by `ocrconfig -restore` is incompatible with the file format generated by `ocrconfig -export`. The `ocrconfig -export` and `ocrconfig -import` commands are compatible. The `ocrconfig -manualbackup` and `ocrconfig -restore` commands are compatible. The two file formats are incompatible and must not be interchangeably used.
- When exporting OCR, Oracle recommends including "`ocr`", the cluster name, and the timestamp in the name string. For example:

  ```
  ocr_mycluster1_20090521_2130_export
  ```

Using the `ocrconfig -export` command also enables you to restore OCR using the `-import` option if your configuration changes cause errors. For example, if you have configuration problems that you cannot resolve, or if you are unable to restart Oracle Clusterware after such changes, then restore your configuration using the procedure for your platform.

Oracle recommends that you use either automatic or manual backups, and the `ocrconfig -restore` command instead of the `ocrconfig -export` and `ocrconfig -import` commands to restore OCR for the following reasons:

- A backup is a consistent snapshot of OCR, whereas an export is not.
- Backups are created when the system is online. You must shut down Oracle Clusterware on all nodes in the cluster to get a consistent snapshot using the `ocrconfig -export` command.
- You can inspect a backup using the OCRDUMP utility. You cannot inspect the contents of an export.
- You can list backups with the `ocrconfig -showbackup` command, whereas *you* must keep track of all generated exports.

This section includes the following topics:

- Importing Oracle Cluster Registry Content on Linux or UNIX Systems

- Importing Oracle Cluster Registry Content on Windows Systems

Note: Most configuration changes that you make not only change OCR contents, the configuration changes also cause file and database object creation. Some of these changes are often not restored when you restore OCR. Do not restore OCR as a correction to revert to previous configurations, if some of these configuration changes should fail. This may result in an OCR location that has contents that do not match the state of the rest of your system.

Importing Oracle Cluster Registry Content on Linux or UNIX Systems

Note: This procedure assumes default installation of Oracle Clusterware on all nodes in the cluster, where Oracle Clusterware autostart is enabled.

Use the following procedure to import OCR on Linux or UNIX systems:

1. List the nodes in your cluster by running the following command on one node:

   ```
   $ olsnodes
   ```

2. Stop Oracle Clusterware by running the following command as `root` on all of the nodes:

   ```
   # crsctl stop crs
   ```

   If the preceding command returns any error due to OCR corruption, stop Oracle Clusterware by running the following command as `root` on all of the nodes:

   ```
   # crsctl stop crs -f
   ```

3. Start the Oracle Clusterware stack on one node in exclusive mode by running the following command as `root`:

   ```
   # crsctl start crs -excl
   ```

   Ignore any errors that display.

   Check whether CRSD is running. If it is, stop it by running the following command as `root`:

   ```
   # crsctl stop resource ora.crsd -init
   ```

   Caution: *Do not* use the `-init` flag with any other command.

4. Import OCR by running the following command as `root`:

   ```
   # ocrconfig -import file_name
   ```

   If you are importing OCR to a cluster or network file system, then skip to step 7.

Notes:

- If the original OCR location does not exist, then you must create an empty (0 byte) OCR location before you run the `ocrconfig -import` command.
- Ensure that the OCR devices that you specify in the OCR configuration exist and that these OCR devices are valid.
- If you configured OCR in an Oracle ASM disk group, then ensure that the Oracle ASM disk group exists and is mounted.

See Also:

- *Oracle Grid Infrastructure Installation Guide* for information about creating OCRs
- *Oracle Automatic Storage Management Administrator's Guide* for more information about Oracle ASM disk group management

5. Verify the integrity of OCR:

```
# ocrcheck
```

6. Stop Oracle Clusterware on the node where it is running in exclusive mode:

```
# crsctl stop crs -f
```

7. Begin to start Oracle Clusterware by running the following command as `root` on all of the nodes:

```
# crsctl start crs
```

8. Verify OCR integrity of all of the cluster nodes that are configured as part of your cluster by running the following CVU command:

```
$ cluvfy comp ocr -n all -verbose
```

Note: You *can only* import an exported OCR. To restore OCR from a backup, you must instead use the `-restore` option, as described in "Backing Up Oracle Cluster Registry" on page 6-10.

See Also: Appendix A, "Cluster Verification Utility Reference" for more information about enabling and using CVU

Importing Oracle Cluster Registry Content on Windows Systems

Note: This procedure assumes default installation of Oracle Clusterware on all nodes in the cluster, where Oracle Clusterware autostart is enabled.

Use the following procedure to import OCR on Windows systems:

1. List the nodes in your cluster by running the following command on one node:

```
C:\>olsnodes
```

2. Stop Oracle Clusterware by running the following command as a member of the Administrators group on all of the nodes:

   ```
   C:\>crsctl stop crs
   ```

   If the preceding command returns any error due to OCR corruption, stop Oracle Clusterware by running the following command as a member of the Administrators group on all of the nodes:

   ```
   C:\>crsctl stop crs -f
   ```

3. Start the Oracle Clusterware stack on one node in exclusive mode by running the following command as a member of the Administrators group:

   ```
   C:\>crsctl start crs -excl
   ```

   Ignore any errors that display.

   Check whether CRSD is running. If it is, stop it by running the following command as a member of the Administrators group:

   ```
   C:\>crsctl stop resource ora.crsd -init
   ```

   Caution: *Do not* use the `-init` flag in any other command.

4. Import OCR by running the following command as a member of the Administrators group:

   ```
   C:\>ocrconfig -import file_name
   ```

   Make sure that the OCR devices that you specify in the OCR configuration exist and that these OCR devices are valid.

   Notes:

   - If the original OCR location does not exist, then you must create an empty (0 byte) OCR location before you run the `ocrconfig -import` command.
   - Ensure that the OCR devices that you specify in the OCR configuration exist and that these OCR devices are valid.
   - Ensure that the Oracle ASM disk group you specify exists and is mounted.

   See Also:

   - *Oracle Grid Infrastructure Installation Guide* for information about creating OCRs
   - *Oracle Automatic Storage Management Administrator's Guide* for more information about Oracle ASM disk group management

5. Verify the integrity of OCR:

   ```
   C:\>ocrcheck
   ```

6. Stop Oracle Clusterware on the node where it is running in exclusive mode:

   ```
   C:\>crsctl stop crs -f
   ```

7. Begin to start Oracle Clusterware by running the following command as a member of the Administrators group on all of the nodes:

   ```
   C:\>crsctl start crs
   ```

8. Run the following Cluster Verification Utility (CVU) command to verify OCR integrity of all of the nodes in your cluster database:

   ```
   C:\>cluvfy comp ocr -n all -verbose
   ```

   > **See Also:** Appendix A, "Cluster Verification Utility Reference" for more information about enabling and using CVU

Oracle Local Registry

In Oracle Clusterware 12*c*, each node in a cluster has a local registry for node-specific resources, called an Oracle Local Registry (OLR), that is installed and configured when Oracle Clusterware installs OCR. Multiple processes on each node have simultaneous read and write access to the OLR particular to the node on which they reside, regardless of whether Oracle Clusterware is running or fully functional.

By default, OLR is located at `Grid_home/cdata/host_name.olr` on each node.

Manage OLR using the OCRCHECK, OCRDUMP, and OCRCONFIG utilities as `root` with the `-local` option.

- You can check the status of OLR on the local node using the OCRCHECK utility, as follows:

  ```
  # ocrcheck -local

  Status of Oracle Cluster Registry is as follows :
          Version                  :          3
          Total space (kbytes)     :     262132
          Used space (kbytes)      :       9200
          Available space (kbytes) :     252932
          ID                       :  604793089
          Device/File Name         : /private2/crs/cdata/localhost/dglnx6.olr
                                     Device/File integrity check succeeded

          Local OCR integrity check succeeded
  ```

- You can display the content of OLR on the local node to the text terminal that initiated the program using the OCRDUMP utility, as follows:

  ```
  # ocrdump -local -stdout
  ```

- You can perform administrative tasks on OLR on the local node using the OCRCONFIG utility.

  – To export OLR to a file:

    ```
    # ocrconfig -local -export file_name
    ```

Notes:

- Oracle recommends that you use the `-manualbackup` and `-restore` commands and not the `-import` and `-export` commands.
- When exporting OLR, Oracle recommends including "`olr`", the host name, and the timestamp in the name string. For example:

  ```
  olr_myhost1_20090603_0130_export
  ```

– To import a specified file to OLR:

```
# ocrconfig -local -import file_name
```

– To manually back up OLR:

```
# ocrconfig -local -manualbackup
```

Note: The OLR is backed up after an installation or an upgrade. After that time, you can only manually back up the OLR. Automatic backups are not supported for the OLR. You should create a new backup when you migrate OCR from Oracle ASM to other storage, or when you migrate OCR from other storage to Oracle ASM.

The default backup location for the OLR is in the path `*Grid_home*/cdata/*host_name*`.

– To view the contents of the OLR backup file:

```
ocrdump -local -backupfile olr_backup_file_name
```

– To change the OLR backup location:

```
ocrconfig -local -backuploc new_olr_backup_path
```

– To restore OLR:

```
# crsctl stop crs
# ocrconfig -local -restore file_name
# ocrcheck -local
# crsctl start crs
$ cluvfy comp olr
```

Upgrading and Downgrading the Oracle Cluster Registry Configuration

When you upgrade Oracle Clusterware, it automatically runs the `ocrconfig -upgrade` command. To downgrade, follow the downgrade instructions for each component and also downgrade OCR using the `ocrconfig -downgrade` command. If you are upgrading OCR, then you can use the OCRCHECK utility to verify the integrity of OCR.

Managing Voting Files

This section includes the following topics for managing voting files in your cluster:

- Storing Voting Files on Oracle ASM
- Backing Up Voting Files

- Restoring Voting Files
- Adding, Deleting, or Migrating Voting Files

Caution: The dd commands used to back up and recover voting files in previous versions of Oracle Clusterware are not supported in Oracle Clusterware 12*c*. Restoring voting files that were copied using dd or cp commands can prevent the Oracle Clusterware 12*c* stack from coming up. Use the backup and restore procedures described in this chapter to ensure proper voting file functionality.

Notes:

- Voting file management requires a valid and working OCR. Before you add, delete, replace, or restore voting files, run the ocrcheck command as root. If OCR is not available or it is corrupt, then you must restore OCR as described in "Restoring Oracle Cluster Registry" on page 6-11.
- If you upgrade from a previous version of Oracle Clusterware to Oracle Clusterware 12*c* and you want to store voting files in an Oracle ASM disk group, then you must set the **ASM Compatibility** (COMPATIBLE.ASM) compatibility attribute to 12.1.0.0.

See Also: *Oracle Automatic Storage Management Administrator's Guide* for information about setting Oracle ASM compatibility attributes

Storing Voting Files on Oracle ASM

Oracle ASM manages voting files differently from other files that it stores. If you choose to store your voting files in Oracle ASM, then Oracle ASM stores all the voting files for the cluster in the disk group you choose. You cannot use voting files stored in Oracle ASM and voting files not stored in Oracle ASM in the same cluster.

Once you configure voting files on Oracle ASM, you can only make changes to the voting files' configuration using the crsctl replace votedisk command. This is true even in cases where there are no working voting files. Despite the fact that crsctl query css votedisk reports zero vote disks in use, Oracle Clusterware remembers the fact that Oracle ASM was in use and the replace verb is required. Only after you use the replace verb to move voting files back to non-Oracle ASM storage are the verbs add css votedisk and delete css votedisk again usable.

The number of voting files you can store in a particular Oracle ASM disk group depends upon the redundancy of the disk group.

By default, Oracle ASM puts each voting file in its own failure group within the disk group. A failure group is a subset of the disks in a disk group. Failure groups define disks that share components, such that if one fails then other disks sharing the component might also fail. An example of what you might define as a failure group would be a set of SCSI disks sharing the same SCSI controller. Failure groups are used to determine which Oracle ASM disks to use for storing redundant data. For example, if two-way mirroring is specified for a file, then redundant copies of file extents must be stored in separate failure groups.

The redundancy level that you choose for the Oracle ASM disk group determines how Oracle ASM mirrors files in the disk group, and determines the number of disks and amount of disk space that you require. If the voting files are in a disk group, then the disk groups that contain Oracle Clusterware files (OCR and voting files) have a higher minimum number of failure groups than other disk groups because the voting files are stored in quorum failure groups.

A quorum failure group is a special type of failure group that is used to store the Oracle Clusterware voting files. The quorum failure group is used to ensure that a quorum of the specified failure groups are available. When Oracle ASM mounts a disk group that contains Oracle Clusterware files, the quorum failure group is used to determine if the disk group can be mounted if there is a loss of one or more failure groups. Disks in the quorum failure group do not contain user data, therefore a quorum failure group is not considered when determining redundancy requirements in respect to storing user data.

Redundancy levels include:

- **External redundancy**: An external redundancy disk group requires a minimum of one disk device. The effective disk space in an external redundancy disk group is the sum of the disk space in all of its devices.

  Because Oracle ASM does not mirror data in an external redundancy disk group, Oracle recommends that you use external redundancy with storage devices such as RAID, or other similar devices that provide their own data protection mechanisms.

- **Normal redundancy**: A normal redundancy disk group requires a minimum of two disk devices (or two failure groups). The effective disk space in a normal redundancy disk group is half the sum of the disk space in all of its devices.

  For Oracle Clusterware files, a normal redundancy disk group requires a minimum of three disk devices (two of the three disks are used by failure groups and all three disks are used by the quorum failure group) and provides three voting files and one OCR and mirror of the OCR. When using a normal redundancy disk group, the cluster can survive the loss of one failure group.

- **High redundancy**: In a high redundancy disk group, Oracle ASM uses three-way mirroring to increase performance and provide the highest level of reliability. A high redundancy disk group requires a minimum of three disk devices (or three failure groups). The effective disk space in a high redundancy disk group is one-third the sum of the disk space in all of its devices.

  For Oracle Clusterware files, a high redundancy disk group requires a minimum of five disk devices (three of the five disks are used by failure groups and all five disks are used by the quorum failure group) and provides five voting files and one OCR and two mirrors of the OCR. With high redundancy, the cluster can survive the loss of two failure groups.

Using the `crsctl replace votedisk` command, you can move a given set of voting files from one Oracle ASM disk group into another, or onto a certified file system. If you move voting files from one Oracle ASM disk group to another, then you can change the number of voting files by placing them in a disk group of a different redundancy level as the former disk group.

Notes:

- You cannot directly influence the number of voting files in one disk group.
- You cannot use the `crsctl add | delete votedisk` commands on voting files stored in Oracle ASM disk groups because Oracle ASM manages the number of voting files according to the redundancy level of the disk group.
- You cannot add a voting file to a cluster file system if the voting files are stored in an Oracle ASM disk group. Oracle does not support having voting files in Oracle ASM and directly on a cluster file system for the same cluster at the same time.

See Also:

- *Oracle Automatic Storage Management Administrator's Guide* for more information about disk group redundancy
- "Adding, Deleting, or Migrating Voting Files" on page 6-26 for information about migrating voting files

Backing Up Voting Files

Oracle Clusterware automatically backs up voting file data in OCR as part of any configuration change and automatically restores the data to any voting file added. If all voting files are corrupted, however, you can restore them as described in "Restoring Voting Files" on page 6-25.

Restoring Voting Files

If all of the voting files are corrupted, then you can restore them, as follows:

1. Restore OCR as described in "Restoring Oracle Cluster Registry" on page 6-11, if necessary.

   This step is necessary only if OCR is also corrupted or otherwise unavailable, such as if OCR is on Oracle ASM and the disk group is no longer available.

   > **See Also:** *Oracle Automatic Storage Management Administrator's Guide* for more information about managing Oracle ASM disk groups

2. Run the following command as `root` from only one node to start the Oracle Clusterware stack in exclusive mode, which does not require voting files to be present or usable:

   ```
   # crsctl start crs -excl
   ```

3. Run the `crsctl query css votedisk` command to retrieve the list of voting files currently defined, similar to the following:

   ```
   $ crsctl query css votedisk
   --  -----    -----------------                --------- ---------
   ##  STATE    File Universal Id                File Name Disk group
    1. ONLINE   6f57843d89464c46ea747362e8a3fa43 (/dev/sdb1) [DATA]
    2. ONLINE   7c54856e98474f61bf349401e7c9fb95 (/dev/sdc1) [DATA]
    3. ONLINE   9c46232b76234f61fc934673d5c8ec59 (/dev/sdd1) [DATA]
   ```

This list may be empty if all voting files were corrupted, or may have entries that are marked as status `3` or `OFF`.

4. Depending on where you store your voting files, do one of the following:

   - If the voting files are stored in Oracle ASM, then run the following command to migrate the voting files to the Oracle ASM disk group you specify:

     ```
     crsctl replace votedisk +asm_disk_group
     ```

     The Oracle ASM disk group to which you migrate the voting files must exist in Oracle ASM. You can use this command whether the voting files were stored in Oracle ASM or some other storage device.

   - If you did not store voting files in Oracle ASM, then run the following command using the File Universal Identifier (FUID) obtained in the previous step:

     ```
     $ crsctl delete css votedisk FUID
     ```

     Add a voting file, as follows:

     ```
     $ crsctl add css votedisk path_to_voting_disk
     ```

5. Stop the Oracle Clusterware stack as `root`:

   ```
   # crsctl stop crs
   ```

   > **Note:** If the Oracle Clusterware stack is running in exclusive mode, then use the `-f` option to force the shutdown of the stack.

6. Restart the Oracle Clusterware stack in normal mode as `root`:

   ```
   # crsctl start crs
   ```

Adding, Deleting, or Migrating Voting Files

You can add, remove, and migrate voting files after you install Oracle Clusterware. Note that the commands you use to do this are different, depending on whether your voting files are located in Oracle ASM, or are located in another storage option.

Modifying voting files that are stored in Oracle ASM

- To display the voting file FUID and file path of each current voting file, run the `crsctl query css votedisk` command to display output similar to the following:

  ```
  $ crsctl query css votedisk
  --  -----    -----------------                --------- ---------
  ##  STATE    File Universal Id                File Name Disk group
   1. ONLINE   6f57843d89464c46ea747362e8a3fa43 (/dev/sdb1) [DATA]
   2. ONLINE   7c54856e98474f61bf349401e7c9fb95 (/dev/sdc1) [DATA]
   3. ONLINE   9c46232b76234f61fc934673d5c8ec59 (/dev/sdd1) [DATA]
  ```

  This command returns a disk sequence number, the status of the disk, the FUID, the path of the disk, and the name of the Oracle ASM disk group on which the disk is stored.

- To migrate voting files from Oracle ASM to an alternative storage device, specify the path to the non-Oracle ASM storage device with which you want to replace the Oracle ASM disk group using the following command:

```
$ crsctl replace votedisk path_to_voting_disk
```

You can run this command on any node in the cluster.

- To replace all voting files not stored in Oracle ASM with voting files managed by Oracle ASM in an Oracle ASM disk group, run the following command:

  ```
  $ crsctl replace votedisk +asm_disk_group
  ```

Modifying voting files that are not stored on Oracle ASM:

- To display the voting file FUID and file path of each current voting file, run the following command:

  ```
  $ crsctl query css votedisk
  ##  STATE    File Universal Id                File Name Disk group
  --  -----    -----------------                --------- ---------
   1. ONLINE   7c54856e98474f61bf349401e7c9fb95 (/cfs/host09_vd3) []
  ```

  This command returns a disk sequence number, the status of the disk, the FUID, and the path of the disk and no name of an Oracle ASM disk group.

- To add one or more voting files, run the following command, replacing the *path_to_voting_disk* variable with one or more space-delimited, complete paths to the voting files you want to add:

  ```
  $ crsctl add css votedisk path_to_voting_disk [...]
  ```

- To replace voting file A with voting file B, you must add voting file B, and then delete voting file A. To add a new disk and remove the existing disk, run the following command, replacing the *path_to_voting_diskB* variable with the fully qualified path name of voting file B:

  ```
  $ crsctl add css votedisk path_to_voting_diskB -purge
  ```

  The `-purge` option deletes existing voting files.

- To remove a voting file, run the following command, specifying one or more space-delimited, voting file FUIDs or comma-delimited directory paths to the voting files you want to remove:

  ```
  $ crsctl delete css votedisk {FUID | path_to_voting_disk[...]}
  ```

> **Note:** If the cluster is down and cannot restart due to lost voting files, then you must start CSS in exclusive mode by running the following command, as `root`:
>
> ```
> # crsctl start crs -excl
> ```
>
> After you start CSS in exclusive mode, you can replace the voting file, as follows:
>
> ```
> # crsctl replace votedisk path_to_voting_disk
> ```

Migrating voting files to Oracle ASM

To migrate voting files to Oracle ASM, specify the Oracle ASM disk group name in the following command:

```
$ crsctl replace votedisk +asm_disk_group
```

You can run this command on any node in the cluster.

Verifying the voting file location

After modifying the voting file, verify the voting file location, as follows:

```
$ crsctl query css votedisk
```

> **See Also:** Appendix E, "Oracle Clusterware Control (CRSCTL) Utility Reference" for more information about CRSCTL commands

7 Adding and Deleting Cluster Nodes

This chapter describes how to add nodes to an existing cluster, and how to delete nodes from clusters. This chapter provides procedures for these tasks for Linux, UNIX, and Windows systems.

Notes:

- Unless otherwise instructed, perform all add and delete node steps as the user that installed Oracle Clusterware.
- Oracle recommends that you use the cloning procedure described in Chapter 8, "Cloning Oracle Clusterware" to create clusters.

The topics in this chapter include the following:

- Prerequisite Steps for Adding Cluster Nodes
- Adding and Deleting Cluster Nodes on Linux and UNIX Systems
- Adding and Deleting Cluster Nodes on Windows Systems

Prerequisite Steps for Adding Cluster Nodes

Note: Ensure that you perform the preinstallation tasks listed in *Oracle Grid Infrastructure Installation Guide for Linux* before adding a node to a cluster.

Do not install Oracle Clusterware. The software is copied from an existing node when you add a node to the cluster.

Complete the following steps to prepare nodes to add to the cluster:

1. Make physical connections.

   Connect the nodes' hardware to the network infrastructure of your cluster. This includes establishing electrical connections, configuring network interconnects, configuring shared disk subsystem connections, and so on. See your hardware vendor documentation for details about this step.

2. Install the operating system.

   Install a cloned image of the operating system that matches the operating system on the other nodes in your cluster. This includes installing required service patches, updates, and drivers. See your operating system vendor documentation for details about this process.

Note: Oracle recommends that you use a cloned image. However, if the installation fulfills the installation requirements, then install the operating system according to the vendor documentation.

3. Create Oracle users.

   You must create all Oracle users on the new node that exist on the existing nodes. For example, if you are adding a node to a cluster that has two nodes, and those two nodes have different owners for the Oracle Grid Infrastructure home and the Oracle home, then you must create those owners on the new node, even if you do not plan to install an Oracle home on the new node.

   Note: Perform this step only for Linux and UNIX systems.

   As `root`, create the Oracle users and groups using the same user ID and group ID as on the existing nodes.

4. Ensure that SSH is configured on the node.

   Note: SSH is configured when you install Oracle Clusterware 12*c*. If SSH is not configured, then see *Oracle Grid Infrastructure Installation Guide* for information about configuring SSH.

5. Verify the hardware and operating system installations with the Cluster Verification Utility (CVU).

   After you configure the hardware and operating systems on the nodes you want to add, you can run the following commands to verify that the nodes you want to add are reachable by other nodes in the cluster. You can also use this command to verify user equivalence to all given nodes from the local node, node connectivity among all of the given nodes, accessibility to shared storage from all of the given nodes, and so on.

   a. From the `Grid_home/bin` directory on an existing node, run the CVU command to obtain a detailed comparison of the properties of the reference node with all of the other nodes that are part of your current cluster environment. Replace *`ref_node`* with the name of a node in your existing cluster against which you want CVU to compare the nodes to be added. Specify a comma-delimited list of nodes after the `-n` option. In the following example, *`orainventory_group`* is the name of the Oracle Inventory group, and *`osdba_group`* is the name of the OSDBA group:

```
$ cluvfy comp peer [-refnode ref_node] -n node_list
[-orainv orainventory_group] [-osdba osdba_group] [-verbose]
```

   b. Ensure that the Grid Infrastructure Management Repository has at least an additional 500 MB of space for each node added above four, as follows:

```
$ oclumon manage -get repsize
```

   Add additional space, if required, as follows:

```
$ oclumon manage -repos changerepossize total_in_MB
```

See Also: "OCLUMON Command Reference" on page J-5 for more information about using OCLUMON

Note: For the reference node, select a cluster node against which you want CVU to compare, for example, the nodes that you want to add that you specify with the `-n` option.

After completing the procedures in this section, you are ready to add the nodes to the cluster.

Note: Avoid changing host names after you complete the Oracle Clusterware installation, including adding or deleting domain qualifications. Nodes with changed host names must be deleted from the cluster and added back with the new name.

Adding and Deleting Cluster Nodes on Linux and UNIX Systems

This section explains cluster node addition and deletion on Linux and UNIX systems. The procedure in the section for adding nodes assumes that you have performed the steps in the "Prerequisite Steps for Adding Cluster Nodes" section.

The last step of the node addition process includes extending the Oracle Clusterware home from an Oracle Clusterware home on an existing node to the nodes that you want to add.

This section includes the following topics:

- Adding a Cluster Node on Linux and UNIX Systems
- Deleting a Cluster Node on Linux and UNIX Systems

Note: Beginning with Oracle Clusterware 11*g* release 2 (11.2), Oracle Universal Installer defaults to silent mode when adding nodes.

Adding a Cluster Node on Linux and UNIX Systems

This procedure describes how to add a node to your cluster. This procedure assumes that:

- There is an existing cluster with two nodes named `node1` and `node2`
- You are adding a node named `node3` using a virtual node name, `node3-vip`, that resolves to an IP address, if you are not using DHCP and Grid Naming Service (GNS)
- You have successfully installed Oracle Clusterware on `node1` and `node2` in a local (non-shared) home, where *`Grid_home`* represents the successfully installed home

To add a node:

1. Ensure that you have successfully installed Oracle Clusterware on at least one node in your cluster environment. To perform the following procedure, *`Grid_home`* must identify your successfully installed Oracle Clusterware home.

> **See Also:** *Oracle Grid Infrastructure Installation Guide* for Oracle Clusterware installation instructions

2. Verify the integrity of the cluster and node3:

   ```
   $ cluvfy stage -pre nodeadd -n node3 [-fixup] [-verbose]
   ```

   You can specify the -fixup option to attempt to fix the cluster or node if the verification fails.

3. To extend the Oracle Grid Infrastructure home to the node3, navigate to the *Grid_home*/addnode directory on node1 and run the addnode.sh script as the user that installed Oracle Clusterware.

   To run addnode.sh in interactive mode, run addnode.sh from *Grid_home*/addnode.

   You can also run addnode.sh in silent mode for both Oracle Clusterware standard Clusters and Oracle Flex Clusters.

   For an Oracle Clusterware standard Cluster:

   ```
   ./addnode.sh -silent "CLUSTER_NEW_NODES={node3}" "CLUSTER_NEW_VIRTUAL_
     HOSTNAMES={node3-vip}"
   ```

   If you are adding node3 to an Oracle Flex Cluster, then you can specify the node role on the command line, as follows:

   ```
   ./addnode.sh -silent "CLUSTER_NEW_NODES={node3}" "CLUSTER_NEW_VIRTUAL_
     HOSTNAMES={node3-vip}" "CLUSTER_NEW_NODE_ROLES={hub}"
   ```

   > **Note:** Hub Nodes always have VIPs but Leaf Nodes may not. If you use the preceding syntax to add multiple nodes to the cluster, then you can use syntax similar to the following, where node3 is a Hub Node and node4 is a Leaf Node:
   >
   > ```
   > ./addnode.sh -silent "CLUSTER_NEW_NODES={node3,node4}" "CLUSTER_
   > NEW_VIRTUAL_HOSTNAMES={node3-vip,}" "CLUSTER_NEW_NODE_
   > ROLES={hub,leaf}"
   > ```

4. If prompted, then run the orainstRoot.sh script as root to populate the /etc/oraInst.loc file with the location of the central inventory. For example:

   ```
   # /opt/oracle/oraInventory/orainstRoot.sh
   ```

5. If you have an Oracle RAC or Oracle RAC One Node database configured on the cluster and you have a local Oracle home, then do the following to extend the Oracle database home to node3:

   a. Navigate to the *Oracle_home*/addnode directory on node1 and run the addnode.sh script as the user that installed Oracle RAC using the following syntax:

      ```
      $ ./addnode.sh "CLUSTER_NEW_NODES={node3}"
      ```

   b. Run the *Oracle_home*/root.sh script on node3 as root, where *Oracle_home* is the Oracle RAC home.

   If you have a shared Oracle home that is shared using Oracle Automatic Storage Management Cluster File System (Oracle ACFS), then do the following to extend the Oracle database home to node3:

a. Run the *Grid_home*/root.sh script on node3 as root, where *Grid_home* is the Oracle Grid Infrastructure home.

b. Run the following command as the user that installed Oracle RAC from the *Oracle_home*/oui/bin directory on the node you are adding to add the Oracle RAC database home:

```
$ ./runInstaller -attachHome ORACLE_HOME="ORACLE_HOME" "CLUSTER_
NODES={node3}"
LOCAL_NODE="node3" ORACLE_HOME_NAME="home_name" -cfs
```

c. Navigate to the *Oracle_home*/addnode directory on node1 and run the addnode.sh script as the user that installed Oracle RAC using the following syntax:

```
$ ./addnode.sh -noCopy "CLUSTER_NEW_NODES={node3}"
```

Note: Use the -noCopy option because the Oracle home on the destination node is already fully populated with software.

If you have a shared Oracle home on a shared file system that *is not* Oracle ACFS, then you must first create a mount point for the Oracle RAC database home on the target node, mount and attach the Oracle RAC database home, and update the Oracle Inventory, as follows:

a. Run the srvctl config database -db *db_name* command on an existing node in the cluster to obtain the mount point information.

b. Run the following command as root on node3 to create the mount point:

```
# mkdir -p mount_point_path
```

c. Mount the file system that hosts the Oracle RAC database home.

d. Run the following command as the user that installed Oracle RAC from the *Oracle_home*/oui/bin directory on the node you are adding to add the Oracle RAC database home:

```
$ ./runInstaller -attachHome ORACLE_HOME="ORACLE_HOME" "CLUSTER
_NODES={local_node_name}" LOCAL_NODE="node_name" ORACLE_HOME_NAME="home_
name" -cfs
```

e. Navigate to the *Oracle_home*/addnode directory on node1 and run the addnode.sh script as the user that installed Oracle RAC using the following syntax:

```
$ ./addnode.sh -noCopy "CLUSTER_NEW_NODES={node3}"
```

Note: Use the -noCopy option because the Oracle home on the destination node is already fully populated with software.

6. Run the *Grid_home*/root.sh script on the node3 as root and run the subsequent script, as instructed.

Notes:

- If you ran the `root.sh` script in the step 5, then you do not need to run it again.
- If you have a policy-managed database, then you must ensure that the Oracle home is cloned to the new node before you run the `root.sh` script.

7. Start the Oracle ACFS resource on the new node by running the following command as `root` from the *`Grid_home`*`/bin` directory:

```
# srvctl start filesystem -device volume_device_name -node node3
```

Note: Ensure the Oracle ACFS resources, including Oracle ACFS registry resource and Oracle ACFS file system resource where the Oracle home is located, are online on the newly added node.

8. Run the following CVU command as the user that installed Oracle Clusterware to check cluster integrity. This command verifies that any number of specified nodes has been successfully added to the cluster at the network, shared storage, and clusterware levels:

```
$ cluvfy stage -post nodeadd -n node3 [-verbose]
```

See Also: `"cluvfy stage [-pre | -post] nodeadd"` on page A-40 for more information about this CVU command

Check whether either a policy-managed or administrator-managed Oracle RAC database is configured to run on `node3` (the newly added node). If you configured an administrator-managed Oracle RAC database, you may need to use DBCA to add an instance to the database to run on this newly added node.

See Also:

- *Oracle Real Application Clusters Administration and Deployment Guide* for more information about using DBCA to add administrator-managed Oracle RAC database instances
- *Oracle Real Application Clusters Administration and Deployment Guide* to add an Oracle RAC database instance to the target node if you configured a policy-managed Oracle RAC database

Deleting a Cluster Node on Linux and UNIX Systems

This section describes the procedure for deleting a node from a cluster.

Notes:

- You can remove the Oracle RAC database instance from the node before removing the node from the cluster but this step is not required. If you do not remove the instance, then the instance is still configured but never runs. Deleting a node from a cluster does not remove a node's configuration information from the cluster. The residual configuration information does not interfere with the operation of the cluster.

  See Also: *Oracle Real Application Clusters Administration and Deployment Guide* for more information about deleting an Oracle RAC database instance

- If you delete the last node of a cluster that is serviced by GNS, then you must delete the entries for that cluster from GNS.
- If you have nodes in the cluster that are unpinned, then Oracle Clusterware ignores those nodes after a time and there is no need for you to remove them.
- If one creates node-specific configuration for a node (such as disabling a service on a specific node, or adding the node to the candidate list for a server pool) that node-specific configuration is not removed when the node is deleted from the cluster. Such node-specific configuration must be removed manually.
- Voting files are automatically backed up in OCR after any changes you make to the cluster.
- When you want to delete a Leaf Node from an Oracle Flex Cluster, you need only complete steps 1 through 4 of this procedure.

To delete a node from a cluster:

1. Ensure that *Grid_home* correctly specifies the full directory path for the Oracle Clusterware home on each node, where *Grid_home* is the location of the installed Oracle Clusterware software.
2. Run the following command as either `root` or the user that installed Oracle Clusterware to determine whether the node you want to delete is active and whether it is pinned:

   ```
   $ olsnodes -s -t
   ```

   If the node is pinned, then run the `crsctl unpin css` command. Otherwise, proceed to the next step.
3. On the node *you want to delete*, run the following command as the user that installed Oracle Clusterware from the *Grid_home*`/oui/bin` directory where *node_to_be_deleted* is the name of the node that you are deleting:

   ```
   $ ./runInstaller -updateNodeList ORACLE_HOME=Grid_home "CLUSTER_NODES=
   {node_to_be_deleted}" CRS=TRUE -silent -local
   ```

4. On the node that you are deleting, depending on whether you have a shared or local Oracle home, complete one of the following procedures as the user that installed Oracle Clusterware:

- For a local home, deinstall the Oracle Clusterware home from the node that you want to delete, as follows, by running the following command, where *Grid_home* is the path defined for the Oracle Clusterware home:

  ```
  $ Grid_home/deinstall/deinstall -local
  ```

  Caution:

  - If you do not specify the `-local` flag, then the command removes the Oracle Grid Infrastructure home from every node in the cluster.
  - If you cut and paste the preceding command, then paste it into a text editor before pasting it to the command line to remove any formatting this document might contain.

- If you have a shared home, then run the following commands in the following order on the node you want to delete.

  Run the following command to deconfigure Oracle Clusterware:

  ```
  $ Grid_home/perl/bin/perl Grid_home/crs/install/rootcrs.pl -deconfig -force
  ```

  Run the following command from the *Grid_home*/`oui/bin` directory to detach the Grid home:

  ```
  $ ./runInstaller -detachHome ORACLE_HOME=Grid_home -silent -local
  ```

  Manually delete any configuration files, as prompted by the installation utility.

5. On any node other than the node you are deleting (except for a Leaf Node in an Oracle Flex Cluster), run the following command from the *Grid_home*/`oui/bin` directory where *`remaining_nodes_list`* is a comma-delimited list of the nodes that are going to remain part of your cluster:

   ```
   $ ./runInstaller -updateNodeList ORACLE_HOME=Grid_home "CLUSTER_NODES=
   {remaining_nodes_list}" CRS=TRUE -silent
   ```

   Notes:

   - You must run this command a second time from the Oracle RAC home, where `ORACLE_HOME=`*`ORACLE__RAC_HOME`* and `CRS=TRUE -silent` is omitted from the syntax, as follows:

     ```
     $ ./runInstaller -updateNodeList ORACLE_HOME=ORACLE_HOME
      "CLUSTER_NODES={remaining_nodes_list}"
     ```

   - Because you do not have to run this command if you are deleting a Leaf Node from an Oracle Flex Cluster, *`remaining_nodes_list`* must list only Hub Nodes.
   - If you have a shared Oracle Grid Infrastructure home, then append the `-cfs` option to the command example in this step and provide a complete path location for the cluster file system.

6. From any node that you are *not deleting*, run the following command from the *Grid_home*/`bin` directory as `root` to delete the node from the cluster:

   ```
   # crsctl delete node -n node_to_be_deleted
   ```

7. Run the following CVU command to verify that the specified nodes have been successfully deleted from the cluster:

```
$ cluvfy stage -post nodedel -n node_list [-verbose]
```

> **See Also:** "cluvfy stage -post nodedel" on page A-41 for more information about this CVU command

8. If you remove a cluster node on which Oracle Clusterware is down, then determine whether the VIP for the deleted node still exists, as follows:

```
$ srvctl config vip -node deleted_node_name
```

If the VIP still exists, then delete it, as follows:

```
$ srvctl stop vip -node deleted_node_name
$ srvctl remove vip -node deleted_node_name
```

Adding and Deleting Cluster Nodes on Windows Systems

This section explains cluster node addition and deletion on Windows systems. This section includes the following topics:

- Adding a Node to a Cluster on Windows Systems
- Deleting a Cluster Node on Windows Systems

> **See Also:** *Oracle Grid Infrastructure Installation Guide* for more information about deleting an entire cluster

Adding a Node to a Cluster on Windows Systems

Ensure that you complete the prerequisites listed in "Prerequisite Steps for Adding Cluster Nodes" on page 7-1 before adding nodes.

This procedure describes how to add a node to your cluster. This procedure assumes that:

- There is an existing cluster with two nodes named `node1` and `node2`
- You are adding a node named `node3`
- You have successfully installed Oracle Clusterware on `node1` and `node2` in a local home, where *`Grid_home`* represents the successfully installed home

> **Note:** Do not use the procedures described in this section to add cluster nodes in configurations where the Oracle database has been upgraded from Oracle Database 10*g* release 1 (10.1) on Windows systems.

To add a node:

1. Verify the integrity of the cluster and `node3`:

```
C:\>cluvfy stage -pre nodeadd -n node3 [-fixup] [-verbose]
```

You can specify the `-fixup` option and a directory into which CVU prints instructions to fix the cluster or node if the verification fails.

2. On node1, go to the *Grid_home*\addnode directory and run the addnode.bat script, as follows:

   ```
   C:\>addnode.bat "CLUSTER_NEW_NODES={node3}"
   "CLUSTER_NEW_VIRTUAL_HOSTNAMES={node3-vip}"
   ```

3. Run the following command on the new node:

   ```
   C:\>Grid_home\crs\config\gridconfig.bat
   ```

4. The following steps are required *only* if you have database homes configured to use Oracle ACFS:

   a. For each database configured to use Oracle ACFS, run the following command from the Oracle RAC database home:

      ```
      C:\> ORACLE_HOME/bin/srvctl stop database -db database_unique_name
      ```

      > **Note:** Run the srvctl config database command to list all of the databases configured with Oracle Clusterware. Use the srvctl config database -db *database_unique_name* to find the database details. If the *ORACLE_HOME* path leads to the Oracle ACFS mount path, then the database uses Oracle ACFS. Use the command output to find the database instance name configured to run on the newly added node.

   b. Use Windows Server Manager Control to stop and delete services.

   c. For each of the databases and database homes collected in step a, run the following command:

      ```
      C:\> ORACLE_HOME/bin/srvctl start database -db database_unique_name
      ```

5. Run the following command to verify the integrity of the Oracle Clusterware components on all of the configured nodes, both the preexisting nodes and the nodes that you have added:

   ```
   C:\>cluvfy stage -post crsinst -n all [-verbose]
   ```

After you complete the procedure in this section for adding nodes, you can optionally extend Oracle Database with Oracle RAC components to the new nodes, making them members of an existing Oracle RAC database.

> **See Also:** *Oracle Real Application Clusters Administration and Deployment Guide* for more information about extending Oracle Database with Oracle RAC to new nodes

Creating the OraMTS Service for Microsoft Transaction Server

Oracle Services for Microsoft Transaction Server (OraMTS) permit Oracle databases to be used as resource managers in Microsoft application-coordinated transactions. OraMTS acts as a proxy for the Oracle database to the Microsoft Distributed Transaction Coordinator (MSDTC). As a result, OraMTS provides client-side connection pooling and allows client components that leverage Oracle to participate in promotable and distributed transactions. In addition, OraMTS can operate with Oracle databases running on any operating system, given that the services themselves are run on Windows.

On releases earlier than Oracle Database 12*c*, the OraMTS service was created as part of a software-only installation. Starting with Oracle Database 12*c*, you must use a configuration tool to create this service.

Create the OraMTS service after adding a node or performing a software-only installation for Oracle RAC, as follows:

1. Open a command window.
2. Change directories to `%ORACLE_HOME%\bin`.
3. Run the OraMTSCtl utility to create the OraMTS Service, where *`host_name`* is a list of nodes on which the service should be created:

   ```
   C:\..bin> oramtsctl.exe -new -host host_name
   ```

   See Also: *Oracle Services for Microsoft Transaction Server Developer's Guide for Microsoft Windows* for more information about OraMTS, which allows Oracle databases to be used as resource managers in distributed transactions

Deleting a Cluster Node on Windows Systems

This section describes how to delete a cluster node on Windows systems. This procedure assumes that Oracle Clusterware is installed on `node1`, `node2`, and `node3`, and that you are deleting `node3` from the cluster.

Notes:

- Oracle does not support using Oracle Enterprise Manager to delete nodes on Windows systems.
- If you delete the last node of a cluster that is serviced by GNS, then you must delete the entries for that cluster from GNS.
- You can remove the Oracle RAC database instance from the node before removing the node from the cluster but this step is not required. If you do not remove the instance, then the instance is still configured but never runs. Deleting a node from a cluster does not remove a node's configuration information from the cluster. The residual configuration information does not interfere with the operation of the cluster.

  See Also: *Oracle Real Application Clusters Administration and Deployment Guide* for more information about deleting an Oracle RAC database instance

To delete a cluster node on Windows systems:

1. Only if you have a local home, on the node you want to delete, run the following command with `-local` option to update the node list:

   ```
   C:\>Grid_home\oui\bin\setup.exe -updateNodeList ORACLE_HOME=Grid_home "CLUSTER_
   NODES=
   {node_to_be_deleted}" CRS=TRUE -local
   ```

   Note: If you are deleting a Leaf Node from an Oracle Flex Cluster, then you do not have run this command.

2. Run the deinstall tool on the node you want to delete to deinstall and deconfigure the Oracle Clusterware home, as follows:

   ```
   C:\Grid_home\deinstall\>deinstall.bat -local
   ```

   Caution:

   - If you do not specify the `-local` flag, then the command removes the Oracle Grid Infrastructure home from every node in the cluster.
   - If you cut and paste the preceding command, then paste it into a text editor before pasting it to the command line to remove any formatting this document might contain.

3. On any node that you are *not* deleting, run the following command from the `Grid_home\oui\bin` directory where `remaining_nodes_list` is a comma-delimited list of the nodes that are going to remain part of your cluster:

   ```
   C:\>setup.exe -updateNodeList ORACLE_HOME=Grid_home
   "CLUSTER_NODES={remaining_nodes_list}" CRS=TRUE -silent
   ```

   Notes:

   - You must run this command a second time where `ORACLE_HOME=ORACLE_HOME`, and `CRS=TRUE -silent` is omitted from the syntax, as follows:

     ```
     C:\>setup.exe -updateNodeList ORACLE_HOME=ORACLE_HOME
      "CLUSTER_NODES={remaining_nodes_list}"
     ```

   - If you have a shared Oracle Grid Infrastructure home, then append the `-cfs` option to the command example in this step and provide a complete path location for the cluster file system.

4. On a node that you are not deleting, run the following command:

   ```
   C:\>Grid_home\bin\crsctl delete node -n node_to_be_deleted
   ```

5. Run the following CVU command to verify that the specified nodes have been successfully deleted from the cluster:

   ```
   C:\>cluvfy stage -post nodedel -n node_list [-verbose]
   ```

   See Also: `"cluvfy stage -post nodedel"` on page A-41

6. If you remove a cluster node on which Oracle Clusterware is down, then determine whether the VIP for the deleted node still exists, as follows:

   ```
   C:\> ORACLE_HOME/bin/srvctl config vip -node deleted_node_name
   ```

   If the VIP still exists, then delete it, as follows:

   ```
   C:\> ORACLE_HOME/bin/srvctl srvctl stop vip -node deleted_node_name
   C:\> ORACLE_HOME/bin/srvctl srvctl remove vip -node deleted_node_name
   ```

8 Cloning Oracle Clusterware

This chapter describes how to clone an Oracle Grid Infrastructure home and use the cloned home to create a cluster. You perform the cloning procedures in this chapter by running scripts in silent mode. The cloning procedures are applicable to Linux and UNIX systems. Although the examples in this chapter use Linux and UNIX commands, the cloning concepts and procedures apply generally to all platforms.

Note: This chapter assumes that you are cloning an Oracle Clusterware 12*c* installation configured as follows:

- No Grid Naming Service (GNS)
- No Intelligent Platform Management Interface specification (IPMI)
- Voting file and Oracle Cluster Registry (OCR) are stored in Oracle Automatic Storage Management (ASM)
- Single Client Access Name (SCAN) resolves through DNS

This chapter contains the following topics:

Introduction to Cloning Oracle Clusterware

Cloning is the process of copying an existing Oracle Clusterware installation to a different location and then updating the copied installation to work in the new environment. Changes made by one-off patches applied on the source Oracle Grid Infrastructure home are also present after cloning. During cloning, you run a script that replays the actions that installed the Oracle Grid Infrastructure home.

Cloning requires that you start with a successfully installed Oracle Grid Infrastructure home. You use this home as the basis for implementing a script that extends the Oracle Grid Infrastructure home to create a cluster based on the original Grid home.

Manually creating the cloning script can be error prone because you prepare the script without interactive checks to validate your input. Despite this, the initial effort is worthwhile for scenarios where you run a single script to configure tens or even hundreds of clusters. If you have only one cluster to install, then you should use the traditional, automated and interactive installation methods, such as Oracle Universal Installer (OUI) or the Provisioning Pack feature of Oracle Enterprise Manager.

> **Note:** Cloning is not a replacement for Oracle Enterprise Manager cloning that is a part of the Provisioning Pack. During Oracle Enterprise Manager cloning, the provisioning process simplifies cloning by interactively asking for details about the Oracle home. The interview questions cover such topics as the location to which you want to deploy the cloned environment, the name of the Oracle database home, a list of the nodes in the cluster, and so on.

The Provisioning Pack feature of Oracle Enterprise Manager Grid Control provides a framework that automates the provisioning of nodes and clusters. For data centers with many clusters, the investment in creating a cloning procedure to provision new clusters and new nodes to existing clusters is worth the effort.

The following list describes some situations in which cloning is useful:

- Cloning prepares an Oracle Grid Infrastructure home once and deploys it to many hosts simultaneously. You can complete the installation in silent mode, as a noninteractive process. You do not need to use a graphical user interface (GUI) console, and you can perform cloning from a Secure Shell (SSH) terminal session, if required.
- Cloning enables you to create an installation (copy of a production, test, or development installation) with all patches applied to it in a single step. Once you have performed the base installation and applied all patch sets and patches on the source system, cloning performs all of these individual steps as a single procedure. This is in contrast to going through the installation process to perform the separate steps to install, configure, and patch the installation on each node in the cluster.
- Installing Oracle Clusterware by cloning is a quick process. For example, cloning an Oracle Grid Infrastructure home to a cluster with more than two nodes requires a few minutes to install the Oracle software, plus a few minutes more for each node (approximately the amount of time it takes to run the `root.sh` script).
- Cloning provides a guaranteed method of accurately repeating the same Oracle Clusterware installation on multiple clusters.

A cloned installation acts the same as its source installation. For example, you can remove the cloned Oracle Grid Infrastructure home using OUI or patch it using OPatch. You can also use the cloned Oracle Grid Infrastructure home as the source for another cloning operation. You can create a cloned copy of a test, development, or production installation by using the command-line cloning scripts.

The default cloning procedure is adequate for most cases. However, you can also customize some aspects of cloning, such as specifying custom port assignments or preserving custom settings.

For example, you can specify a custom port for the listener, as follows:

```
$ export ORACLE_HOME=/u01/app/12.1.0/grid
$ $ORACLE_HOME/bin/srvctl modify listener -endpoints tcp:12345
```

The cloning process works by copying all of the files from the source Oracle Grid Infrastructure home to the destination Oracle Grid Infrastructure home. You can clone either a local (non-shared) or shared Oracle Grid Infrastructure home. Thus, any files used by the source instance that are located outside the source Oracle Grid Infrastructure home's directory structure are not copied to the destination location.

The size of the binary files at the source and the destination may differ because these files are relinked as part of the cloning operation, and the operating system patch levels may also differ between these two locations. Additionally, the number of files in the cloned home would increase because several files copied from the source, specifically those being instantiated, are backed up as part of the clone operation.

Preparing the Oracle Grid Infrastructure Home for Cloning

To prepare the source Oracle Grid Infrastructure home to be cloned, create a copy of an installed Oracle Grid Infrastructure home and then use it to perform the cloning procedure on other nodes. Use the following step-by-step procedure to prepare the copy of the Oracle Grid Infrastructure home:

- Step 1: Install Oracle Clusterware
- Step 2: Shut Down Running Software
- Step 3: Create a Copy of the Oracle Grid Infrastructure Home

Step 1: Install Oracle Clusterware

Use the detailed instructions in the *Oracle Grid Infrastructure Installation Guide* to perform the following steps on the source node:

1. Install Oracle Clusterware 12*c*. This installation puts Oracle Cluster Registry (OCR) and the voting file on Oracle Automatic Storage Management (Oracle ASM).

   > **Note:** Either install and configure the Oracle Grid Infrastructure for a cluster or install just the Oracle Clusterware software, as described in your platform-specific *Oracle Grid Infrastructure Installation Guide*.

   If you installed and configured Oracle Grid Infrastructure for a cluster, then you must stop Oracle Clusterware before performing the cloning procedures. If you performed a software-only installation, then you do not have to stop Oracle Clusterware.

2. Install any patches that are required (for example, an Oracle Grid Infrastructure bundle patch), if necessary.

3. Apply one-off patches, if necessary.

   See Also: *Oracle Grid Infrastructure Installation Guide* for Oracle Clusterware installation instructions

Step 2: Shut Down Running Software

Before copying the source Oracle Grid Infrastructure home, shut down all of the services, databases, listeners, applications, Oracle Clusterware, and Oracle ASM instances that run on the node. Oracle recommends that you use the Server Control (SRVCTL) utility to first shut down the databases, and then the Oracle Clusterware Control (CRSCTL) utility to shut down the rest of the components.

See Also:

- Appendix E, "Oracle Clusterware Control (CRSCTL) Utility Reference" for more information about CRSCTL
- *Oracle Real Application Clusters Administration and Deployment Guide* for more information about SRVCTL

Step 3: Create a Copy of the Oracle Grid Infrastructure Home

To keep the installed Oracle Grid Infrastructure home as a working home, make a full copy of the source Oracle Grid Infrastructure home for cloning.

Tip: When creating the copy, a best practice is to include the release number in the name of the file.

Use one of the following methods to create a compressed copy of the Oracle Grid Infrastructure home, where *`Grid_home`* is the original Oracle Grid Infrastructure home on the original node with all files included, and *`copy_path`* is the directory path to the copied Oracle Grid Infrastructure home with unnecessary files deleted.

Method 1: Create a copy of the Oracle Grid Infrastructure home and remove the unnecessary files from the copy:

1. On the source node, create a copy of the Oracle Grid Infrastructure home. To keep the installed Oracle Grid Infrastructure home as a working home, make a full copy of the source Oracle Grid Infrastructure home and remove the unnecessary files from the copy. For example, as `root` on Linux systems, run the `cp` command:

```
# cp -prf Grid_home copy_path
```

2. Delete unnecessary files from the copy.

   The Oracle Grid Infrastructure home contains files that are relevant only to the source node, so you can remove the unnecessary files from the copy of the Oracle Grid Infrastructure home in the `log`, `crs/init`, `crf`, and `cdata` directories. The following example for Linux and UNIX systems shows the commands to run to remove the unnecessary files from the *copy* of the Oracle Grid Infrastructure home:

```
[root@node1 root]# cd copy_path
[root@node1 grid]# rm -rf log/host_name
[root@node1 grid]# rm -rf gpnp/host_name
[root@node1 grid]# find gpnp -type f -exec rm -f {} \;
[root@node1 grid]# rm -rf cfgtoollogs/*
[root@node1 grid]# rm -rf crs/init/*
[root@node1 grid]# rm -rf cdata/*
[root@node1 grid]# rm -rf crf/*
[root@node1 grid]# rm -rf network/admin/*.ora
[root@node1 grid]# rm -rf crs/install/crsconfig_params
[root@node1 grid]# find . -name '*.ouibak' -exec rm {} \;
[root@node1 grid]# find . -name '*.ouibak.1' -exec rm {} \;
[root@node1 grid]# rm -rf root.sh*
[root@node1 grid]# rm -rf rdbms/audit/*
[root@node1 grid]# rm -rf rdbms/log/*
[root@node1 grid]# rm -rf inventory/backup/*
```

3. Create a compressed copy of the previously copied Oracle Grid Infrastructure home using tar or gzip on Linux and UNIX systems. Ensure that the tool you use preserves the permissions and file timestamps. For example:

On Linux and UNIX systems:

```
[root@node1 root]# cd copy_path
[root@node1 grid]# tar -zcvpf /copy_path/gridHome.tgz .
```

In the preceding example, the `cd` command changes the location to the copy of the Oracle Grid Infrastructure home with the unnecessary files removed that you created in the first two steps of this procedure, and the `tar` command creates a file named `gridHome.tgz`. In the `tar` command, *`copy_path`* represents the location of the copy of the Oracle Grid Infrastructure home.

On AIX or HPUX systems:

```
uncompress gridHome.tar.Z
tar xvf gridHome.tar
```

On Windows systems, use WinZip to create a zip file.

Method 2: Create a compressed copy of the Oracle Grid Infrastructure home using the -X option:

1. Create a file that lists the unnecessary files in the Oracle Grid Infrastructure home. For example, list the following file names, using the asterisk (*) wildcard, in a file called `excludeFileList`:

```
Grid_home/host_name
Grid_home/log/host_name
Grid_home/gpnp/host_name
Grid_home/crs/init/*
Grid_home/cdata/*
Grid_home/crf/*
Grid_home/network/admin/*.ora
Grid_home/root.sh*
*.ouibak
*.ouibak1
```

2. Use the `tar` command or Winzip to create a compressed copy of the Oracle Grid Infrastructure home. For example, on Linux and UNIX systems, run the following command to archive and compress the source Oracle Grid Infrastructure home:

```
tar cpfX - excludeFileList Grid_home | compress -fv > temp_dir/gridHome.tar.Z
```

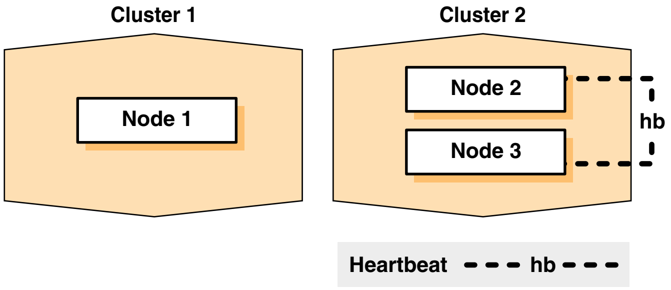
> **Note:** Do not use the jar utility to copy and compress the Oracle Grid Infrastructure home.

Creating a Cluster by Cloning Oracle Clusterware

This section explains how to create a cluster by cloning a successfully installed Oracle Clusterware environment and copying it to the nodes on the destination cluster. The procedures in this section describe how to use cloning for Linux, UNIX, and Windows systems. OCR and voting files are not shared between the two clusters after you successfully create a cluster from a clone.

For example, you can use cloning to quickly duplicate a successfully installed Oracle Clusterware environment to create a cluster. Figure 8–1 shows the result of a cloning procedure in which the Oracle Grid Infrastructure home on Node 1 has been cloned to Node 2 and Node 3 on Cluster 2, making Cluster 2 a new two-node cluster.

Figure 8–1 Cloning to Create an Oracle Clusterware Environment



The steps to create a cluster through cloning are as follows:

- Step 1: Prepare the New Cluster Nodes
- Step 2: Deploy the Oracle Grid Infrastructure Home on the Destination Nodes
- Step 3: Run the clone.pl Script on Each Destination Node
- Step 4: Launch the Configuration Wizard

Step 1: Prepare the New Cluster Nodes

On each destination node, perform the following preinstallation steps:

- Specify the kernel parameters
- Configure block devices for Oracle Clusterware devices
- Ensure that you have set the block device permissions correctly
- Use short, nondomain-qualified names for all of the names in the `/etc/hosts` file
- Test whether the interconnect interfaces are reachable using the `ping` command
- Verify that the VIP addresses are not active at the start of the cloning process by using the `ping` command (the `ping` command of the VIP address must fail)
- Copy the following Oracle Automatic Storage Management Cluster File System tunable files from the source node to each destination node:

  On Linux and UNIX: `/etc/sysconfig/advmtunables` and `/etc/sysconfig/acfstunables`

  On AIX: `/etc/advmtunables` and `/etc/acfstunables`

  On Solaris: `/etc/advmtunables` and `/etc/acfstunables`

  On Windows: `C:\windows\system32\drivers\advm\tunables` and `C:\windows\system32\drivers\acfs\tunables`

- On AIX systems, and on Solaris x86-64-bit systems running vendor clusterware, if you add a node to the cluster, then you must run the `rootpre.sh` script (located at the mount point it you install Oracle Clusterware from a DVD or in the directory where you unzip the tar file if you download the software) on the node before you add the node to the cluster
- Run CVU to verify your hardware and operating system environment

Refer to your platform-specific Oracle Clusterware installation guide for the complete preinstallation checklist.

Note: Unlike traditional methods of installation, the cloning process does not validate your input during the preparation phase. (By comparison, during the traditional method of installation using OUI, various checks occur during the interview phase.) Thus, if you make errors during the hardware setup or in the preparation phase, then the cloned installation fails.

Step 2: Deploy the Oracle Grid Infrastructure Home on the Destination Nodes

Before you begin the cloning procedure that is described in this section, ensure that you have completed the prerequisite tasks to create a copy of the Oracle Grid Infrastructure home, as described in the section titled "Preparing the Oracle Grid Infrastructure Home for Cloning" on page 8-3.

1. On each destination node, deploy the copy of the Oracle Grid Infrastructure home that you created in "Step 3: Create a Copy of the Oracle Grid Infrastructure Home" on page 8-4, as follows:

   If you do not have a shared Oracle Grid Infrastructure home, then restore the copy of the Oracle Grid Infrastructure home on each node in the destination cluster. Use the equivalent directory structure as the directory structure that was used in the Oracle Grid Infrastructure home on the source node. Skip this step if you have a shared Oracle Grid Infrastructure home.

   For example, on Linux or UNIX systems, run commands similar to the following:

   ```
   [root@node1 root]# mkdir -p location_of_the_copy_of_the_Grid_home
   [root@node1 root]# cd location_of_the_copy_of_the_Grid_home
   [root@node1 crs]# tar -zxvf /gridHome.tgz
   ```

   In this example, *location_of_the_copy_of_the_Grid_home* represents the directory structure in which you want to install the Oracle Grid Infrastructure home, such as /u01/app/12.1.0/grid. Note that you can change the Grid home location as part of the clone process.

   On Windows systems, unzip the Oracle Grid Infrastructure home on the destination node in the equivalent directory structure as the directory structure in which the Oracle Grid Infrastructure home resided on the source node.

2. If you have not already deleted unnecessary files from the Oracle Grid Infrastructure home, then repeat step 2 in "Method 1: Create a copy of the Oracle Grid Infrastructure home and remove the unnecessary files from the copy:" on page 8-4.

3. Create a directory for the Oracle Inventory on the destination node and, if necessary, change the ownership of all of the files in the Oracle Grid Infrastructure home to be owned by the Oracle Grid Infrastructure installation owner and by the Oracle Inventory (oinstall privilege) group. If the Oracle Grid Infrastructure installation owner is oracle, and the Oracle Inventory group is oinstall, then the following example shows the commands to do this on a Linux system:

   ```
   [root@node1 crs]# chown -R oracle:oinstall /u01/app
   ```

   When you run the preceding command on the Grid home, it clears setuid and setgid information from the Oracle binary. As expected, the command also clears setuid from the following binaries:

   ```
   Grid_home/bin/extjob
   Grid_home/bin/jssu
   ```

```
Grid_home/bin/oradism
```

The `setuid` information is properly set after you run the `root.sh` script at the end of the cloning procedure.

4. It is important to remove any Oracle network files from the *Grid_home* directory on both the source and destination nodes before continuing.

> **See Also:** "Locating and Viewing Log Files Generated During Cloning" on page 8-13

Step 3: Run the clone.pl Script on Each Destination Node

To set up the new Oracle Clusterware environment, the `clone.pl` script requires you to provide several setup values for the script. You can provide the variable values by either supplying input on the command line when you run the `clone.pl` script, or by creating a file in which you can assign values to the cloning variables. The following discussions describe these options.

> **Note:** After you run `clone.pl`, the script prompts you to run `orainstRoot.sh` and `root.sh`. Run only `orainstRoot.sh` and then proceed to "Step 4: Launch the Configuration Wizard" on page 8-10. The configuration wizard will prompt you to run `root.sh`.

Supplying input to the clone.pl script on the command line

If you do not have a shared Oracle Grid Infrastructure home, navigate to the *Grid_home*/clone/bin directory on each destination node and run the `clone.pl` script, which performs the main Oracle Clusterware cloning tasks. To run the script, you must supply input to several parameters. Table 8–1 describes the `clone.pl` script parameters.

Table 8–1 Parameters for the clone.pl Script

Parameters	Description
`ORACLE_BASE=`*`ORACLE_BASE`*	The complete path to the Oracle base to be cloned. If you specify an invalid path, then the script exits. This parameter is required.
`ORACLE_HOME=`*`GRID_HOME`*	The complete path to the Oracle Grid Infrastructure home for cloning. If you specify an invalid path, then the script exits. This parameter is required.

`[ORACLE_HOME_NAME=`
`Oracle_home_name` `|`
`-defaultHomeName]`
	The Oracle home name of the home to be cloned. Optionally, you can specify the `-defaultHomeName` flag. This parameter is required.

`[ORACLE_HOME_USER=`
`Oracle_home_user_name``]`
	The Oracle home user on Windows. Oracle recommends that you pass this parameter for Oracle Database software cloning. This parameter is optional.

`INVENTORY_LOCATION=`
`location_of_inventory`
	The location for the Oracle Inventory.

`OSDBA_GROUP=`
`OSDBA_privileged_group`
	Specify the operating system group you want to use as the OSDBA privileged group. This parameter is optional if you do not want the default value.

Table 8–1 (Cont.) Parameters for the clone.pl Script

Parameters	Description
"CLUSTER_NODES={*node_name*,*node_name*,...}"	A comma-delimited (with no spaces) list of short node names for the nodes that are included in this new cluster. The following only apply if you are cloning database homes: ■ If you run `clone.pl` on a Hub Node in an Oracle Flex Cluster configuration, then this list must include all the Hub Nodes in the cluster. ■ If you run `clone.pl` on a Leaf Node in an Oracle Flex Cluster configuration, then you must specify only the local host name.
"LOCAL_NODE=*node_name*"	The short node name for the node on which `clone.pl` is running.
CRS=TRUE	This parameter is necessary to set this property on the Oracle Universal Installer inventory.
OSASM_GROUP=*OSASM_privileged_group*	Specify the operating system group you want to use as the OSASM privileged group. This parameter is optional if you do not want the default value.
OSOPER_GROUP=*OSOPER_privileged_group*	Specify the operating system group you want to use as the OSOPER privileged group. This parameter is optional if you do not want the default value.
-debug	Specify this option to run the `clone.pl` script in debug mode.
-help	Specify this option to obtain help for the `clone.pl` script.

For example, on Linux and UNIX systems:

```
$ perl clone.pl -silent ORACLE_BASE=/u01/app/oracle ORACLE_HOME=
/u01/app/12.1/grid ORACLE_HOME_NAME=OraHome1Grid
INVENTORY_LOCATION=/u01/app/oraInventory LOCAL_NODE=node1 CRS=TRUE
```

On Windows systems:

```
C:\>perl clone.pl ORACLE_BASE=D:\u01\app\grid ORACLE_HOME=D:\u01\app\grid\12.1
ORACLE_HOME_NAME=OraHome1Grid ORACLE_HOME_USER=Oracle_home_user_name
"LOCAL_NODE=node1" "CLUSTER_NODES={node1,node2}" CRS=TRUE
```

For Windows platforms, on all other nodes, run the same command with an additional argument: `PERFORM_PARTITION_TASKS=FALSE`.

For example:

```
C:\>perl clone.pl ORACLE_BASE=D:\u01\app\grid ORACLE_HOME=D:\u01\app\grid\12.1
ORACLE_HOME_NAME=OraHome1Grid ORACLE_HOME_USER=Oracle_home_user_name
"LOCAL_NODE=node1" "CLUSTER_NODES={node1,node2}" CRS=TRUE PERFORM_PARTITION_
TASKS=FALSE
```

Refer to Table 8–1 for descriptions of the various parameters in the preceding examples.

If you have a shared Oracle Grid Infrastructure home, then append the `-cfs` option to the command example in this step and provide a complete path location for the cluster file system.

Supplying Input to the clone.pl Script in a File

Because the `clone.pl` script is sensitive to the parameter values that it receives, you must be accurate in your use of brackets, single quotation marks, and double quotation marks. To avoid errors, create a file that is similar to the `start.sh` script shown in Example 8–1 in which you can specify environment variables and cloning parameters for the `clone.pl` script.

Example 8–1 shows an excerpt from an example script called `start.sh` that calls the `clone.pl` script; the example is configured for a cluster named `crscluster`. Run the script as the operating system user that installed Oracle Clusterware.

Example 8–1 Excerpt From the start.sh Script to Clone Oracle Clusterware

```
#!/bin/sh
ORACLE_BASE=/u01/app/oracle
GRID_HOME=/u01/app/12.1/grid
THIS_NODE=`hostname -s`

E01=ORACLE_BASE=${ORACLE_BASE}
E02=ORACLE_HOME=${GRID_HOME}
E03=ORACLE_HOME_NAME=OraGridHome1
E04=INVENTORY_LOCATION=${ORACLE_BASE}/../oraInventory

#C00="-debug"
C01="CLUSTER_NODES={node1,node2}"
C02="LOCAL_NODE=$THIS_NODE"

perl ${GRID_HOME}/clone/bin/clone.pl -silent $E01 $E02 $E03 $E04 $C01 $C02
CRS=TRUE
```

Note: On Solaris systems, there is no `-s` following `hostname`.

The `start.sh` script sets several environment variables and cloning parameters, as described in Table 8–2. Table 8–2 describes the environment variables `E01`, `E02`, `E03`, and `E04` that are shown in bold typeface in Example 8–1, and `C01` and `C02`.

Table 8–2 Environment Variables Passed to the clone.pl Script

Symbol	Variable	Description
E01	ORACLE_BASE	The location of the Oracle base directory.
E02	ORACLE_HOME	The location of the Oracle Grid Infrastructure home. This directory location must exist and must be owned by the Oracle operating system group: OINSTALL.
E03	ORACLE_HOME_NAME	The name of the Oracle Grid Infrastructure home. This is stored in the Oracle Inventory.
E04	INVENTORY_LOCATION	The location of the Oracle Inventory. This directory location must exist and must initially be owned by the Oracle operating system group: OINSTALL.
C01	CLUSTER_NODES	A comma-delimited list of short node names for the nodes in the cluster.
C02	LOCAL_NODE	The short name of the local node.

Step 4: Launch the Configuration Wizard

The Configuration Wizard helps you to prepare the `crsconfig_params` file, prompts you to run the `root.sh` script (which calls the `rootcrs.pl` script), relinks Oracle binaries, and runs cluster verifications.

Start the Configuration Wizard, as follows:

On Linux/UNIX:

```
$ Oracle_home/crs/config/config.sh
```

On Windows:

```
C:\>Oracle_home\crs\config\config.bat
```

Optionally, you can run the Configuration Wizard silently, as follows, providing a response file:

```
$ Oracle_home/crs/config/config.sh -silent -responseFile file_name
```

On Windows:

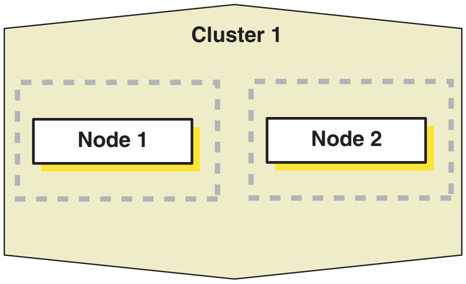
```
C:\>Oracle_home\crs\config\config.bat -silent -responseFile file_name
```

> **See Also:** *Oracle Grid Infrastructure Installation Guide* for your platform for information about preparing response files

Using Cloning to Add Nodes to a Cluster

You can also use cloning to add nodes to a cluster. Figure 8–2 shows the result of a cloning procedure in which the Oracle Grid Infrastructure home on Node 1 has been cloned to Node 2 in the same cluster, making it a two-node cluster. Newly added nodes to the cluster share the same OCR and voting files.

Figure 8–2 Cloning to Add Nodes to a Cluster



Using Figure 8–2 as an example, the following procedure explains how to add nodes to a cluster using cloning. In this procedure, you make a copy of the image (a clone) that you used to create Node 1, initially, to Node 2.

1. Prepare Node 2 as described in "Step 1: Prepare the New Cluster Nodes" on page 8-6.
2. Deploy the Oracle Grid Infrastructure home on Node 2, as described in "Step 2: Deploy the Oracle Grid Infrastructure Home on the Destination Nodes" on page 8-7.

   Use the tar utility to create an archive of the Oracle Grid Infrastructure home on the Node 1 and copy it to Node 2. If the location of the Oracle Grid Infrastructure home on Node 1 is `$ORACLE_HOME`, then you must use this same directory as the destination location on Node 2.
3. Run the `clone.pl` script located in the *`Grid_home`*`/clone/bin` directory on Node 2.

   > **See Also:** Table 8–1 for more information about the parameters used in the `clone.pl` script

   The following example is for Linux or UNIX systems:

   ```
   $ perl clone.pl ORACLE_HOME=/u01/app/12.1/grid ORACLE_HOME_NAME=OraHome1Grid
   ```

```
   ORACLE_BASE=/u01/app/oracle "'CLUSTER_NODES={node1, node2}'"
   "'LOCAL_NODE=node2'" CRS=TRUE INVENTORY_LOCATION=/u01/app/oraInventory
```

If you are prompted to run `root.sh`, then ignore the prompt and proceed to the next step.

The following example is for Windows systems:

```
C:\>perl clone.pl ORACLE_BASE=D:\u01\app\grid ORACLE_
HOME=D:\u01\app\grid\12.1.0
ORACLE_HOME_NAME=OraHome1Grid '"CLUSTER_NODES={node1,node2}"'
'"LOCAL_NODE=node2"' CRS=TRUE
```

Note: In the preceding command, `ORACLE_HOME_NAME` is required when cloning a node to add a node. You can obtain the correct value from node you are cloning from that node's registry, as follows:

```
HKEY_LOCAL_MACHINE\SOFTWARE\oracle\KEY_OraCRs12c_home1
```

Look for the `ORACLE_HOME_NAME` parameter key to obtain the value. If the value for the `ORACLE_HOME_NAME` parameter in the preceding command does not match that of the node you are cloning, then adding the new node will fail.

---

4. This step does not apply to Windows.

   In the Central Inventory directory on Node 2, run the `orainstRoot.sh` script as `root`. This script populates the `/etc/oraInst.loc` directory with the location of the central inventory. For example:

   ```
   [root@node2 root]# /opt/oracle/oraInventory/orainstRoot.sh
   ```

   You can run the script on more than one destination node simultaneously.

5. Run the `addnode.sh` (`addnode.bat` on Windows) script, located in the *`Grid_home`*`/addnode` directory, on Node 1, as follows:

   ```
   $ addnode.sh -silent -noCopy ORACLE_HOME=Grid_home "CLUSTER_NEW_NODES={node2}"
      "CLUSTER_NEW_VIRTUAL_HOSTNAMES={node2-vip}" "CLUSTER_NEW_NODE_ROLES={HUB}"
   ```

Notes:

- Because you already ran the `clone.pl` script on Node 2, this step only updates the inventories on the node and instantiates scripts on the local node.
- If you use the `-noCopy` option with the `addnode.sh` script, then a copy of the password file may not exist on Node 2, in which case you must copy a correct password file to Node 2.
- The `addnode.sh` script runs the `cluvfy stage -pre nodeadd` verification.
- Use the `CLUSTER_NEW_NODE_ROLES` parameter to indicate, in an Oracle Flex Cluster, whether the node you are adding is a Hub Node or a Leaf Node.

You can add multiple nodes, as follows:

```
$ addnode.sh -silent -noCopy ORACLE_HOME=Grid_home "CLUSTER_NEW_
```

```
NODES={node2,node3,node4}"
   "CLUSTER_NEW_VIRTUAL_HOSTNAMES={node2-vip,node3-vip,}"
   "CLUSTER_NEW_NODE_ROLES={HUB,HUB,LEAF}"
```

In the preceding syntax example, Node 4 is designated as a Leaf Node and does not require that a VIP be included.

6. Copy the following files from Node 1, on which you ran `addnode.sh`, to Node 2:

   ```
   Grid_home/crs/install/crsconfig_addparams
   Grid_home/crs/install/crsconfig_params
   Grid_home/gpnp
   ```

7. On Node 2, run the *Grid_home*/`root.sh` script.

   Notes:

   - Ensure that you extend any database homes before you run the `root.sh` or `gridconfig.bat` scripts.
   - The cluster in this example has only two nodes. When you add multiple nodes to a cluster, you can run `root.sh` concurrently on all of the nodes.

   The following example is for a Linux or UNIX system. On Node 2, run the following command:

   ```
   [root@node2 root]# Grid_home/root.sh
   ```

   The `root.sh` script automatically configures the virtual IP (VIP) resources in the Oracle Cluster Registry (OCR).

   On Windows, run the following command on Node 2:

   ```
   C:\>Grid_home\crs\config\gridconfig.bat
   ```

8. Run the following cluster verification utility (CVU) command on Node 1:

   ```
   $ cluvfy stage -post nodeadd -n destination_node_name [-verbose]
   ```

   See Also: "`cluvfy stage [-pre | -post] nodeadd`" on page A-40 for more information about this CVU command

Locating and Viewing Log Files Generated During Cloning

The cloning script runs multiple tools, each of which can generate log files. After the `clone.pl` script finishes running, you can view log files to obtain more information about the status of your cloning procedures. Table 8–3 lists the log files that are generated during cloning that are the key log files for diagnostic purposes.

Note: *`Central_inventory`* in Table 8–3 refers to the Oracle Inventory directory.

Table 8–3 Cloning Log Files and their Descriptions

Log File Name and Location	Description
`Central_inventory/logs/cloneActions/timestamp.log`	Contains a detailed log of the actions that occur during the OUI part of the cloning.
`Central_inventory/logs/oraInstall/timestamp.err`	Contains information about errors that occur when OUI is running.
`Central_inventory/logs/oraInstall/timestamp.out`	Contains other miscellaneous information.

Table 8–4 lists the location of the Oracle Inventory directory for various platforms.

Table 8–4 Finding the Location of the Oracle Inventory Directory

Type of System	Location of the Oracle Inventory Directory
All UNIX computers except Linux and IBM AIX	`/var/opt/oracle/oraInst.loc`
IBM AIX and Linux	`/etc/oraInst.loc`
Windows	`C:\Program Files\Oracle\Inventory`

9

Making Applications Highly Available Using Oracle Clusterware

When an application, process, or server fails in a cluster, you want the disruption to be as short as possible, if not completely unknown to users. For example, when an application fails on a server, that application can be restarted on another server in the cluster, minimizing or negating any disruption in the use of that application. Similarly, if a server in a cluster fails, then all of the applications and processes running on that server must be able to fail over to another server to continue providing service to the users. Using the built-in `generic_application` resource type or customizable scripts and application agent programs, and resource attributes that you assign to applications and processes, Oracle Clusterware can manage all these entities to ensure high availability.

This chapter explains how to use Oracle Clusterware to start, stop, monitor, restart, and relocate applications. Oracle Clusterware is the underlying cluster solution for Oracle Real Application Clusters (Oracle RAC). The same functionality and principles you use to manage Oracle RAC databases are applied to the management of applications.

This chapter includes the following topics:

- Oracle Clusterware Resources and Agents
- Overview of Using Oracle Clusterware to Enable High Availability
- Registering an Application as a Resource
- Managing Resources
- Managing Automatic Restart of Oracle Clusterware Resources

Oracle Clusterware Resources and Agents

This section discusses the framework that Oracle Clusterware uses to monitor and manage **resource**s, to ensure high application availability.

This section includes the following topics:

- Resources
- Resource Types
- Agents
- Action Scripts
- Building an Agent

- Registering a Resource in Oracle Clusterware

Resources

Oracle Clusterware manages applications and processes as resources that you register with Oracle Clusterware. The number of resources you register with Oracle Clusterware to manage an application depends on the application. Applications that consist of only one process are usually represented by only one resource. More complex applications, built on multiple processes or components, may require multiple resources.

When you register an application as a resource in Oracle Clusterware, you define how Oracle Clusterware manages the application using resource attributes you ascribe to the resource. The frequency with which the resource is checked and the number of attempts to restart a resource on the same server after a failure before attempting to start it on another server (failover) are examples of resource attributes. The registration information also includes a path to an action script or application-specific action program that Oracle Clusterware calls to start, stop, check, and clean up the application.

An action script is a shell script (a batch script in Windows) that a generic script agent provided by Oracle Clusterware calls. An application-specific agent is usually a C or C++ program that calls Oracle Clusterware-provided APIs directly.

> **See Also:** Appendix B, "Oracle Clusterware Resource Reference" for an example of an action script

Resource Types

Generally, all resources are unique but some resources may have common attributes. Oracle Clusterware uses resource types to organize these similar resources. Benefits that resource types provide are:

- Manage only necessary resource attributes
- Manage all resources based on the resource type

Every resource that you register in Oracle Clusterware must have a certain resource type. In addition to the resource types included in Oracle Clusterware, you can define custom resource types using the Oracle Clusterware Control (CRSCTL) utility. The included resource types are:

- **Local resource**: Instances of local resources—type name is `local_resource`—run on each server of the cluster (the default) or you can limit them to run on servers belonging to a particular server category. When a server joins the cluster, Oracle Clusterware automatically extends local resources to have instances tied to the new server. When a server leaves the cluster, Oracle Clusterware automatically sheds the instances of local resources that ran on the departing server. Instances of local resources are pinned to their servers; they do not fail over from one server to another.
- **Cluster resource**: Cluster-aware resource types—type name is `cluster_resource`—are aware of the cluster environment and are subject to **cardinality** and cross-server switchover and failover.
- **Generic application**: You can use this resource type—type name is `generic_application`—to protect any generic applications without requiring additional scripts. High availability for an application is achieved by defining a resource with the `generic_application` resource type and providing the values for key attributes of the resource. The `generic_application` resource type is derived from

the cluster_resource resource type and, therefore, all resources of the generic_application resource type are cluster-aware resources. Attributes include:

– START_PROGRAM: A complete path to the executable that starts the application, with all appropriate arguments. The executable must exist on every server where Oracle Grid Infrastructure is configured to run the application. This attribute is required. For example:

```
/opt/my_app -start
```

  The executable must also ensure that the application starts and return an exit status value of zero (0) to indicate that the application started successfully and is online. If the executable fails to start the application, then the executable exits with a non-zero status code.

– STOP_PROGRAM: A complete path to the executable that stops the application, with all appropriate arguments. The executable must exist on every server where Oracle Grid Infrastructure is configured to run the application. If you do not specify this attribute value, then Oracle Clusterware uses an operating system-equivalent of the kill command. For example:

```
/opt/my_app -stop
```

  The executable must also ensure that the application stops and return an exit status value of zero (0) to indicate that the application stopped successfully. If the executable fails to stop the application, then the executable exits with a non-zero status code and Oracle Clusterware initiates a clean of the resource.

– CLEAN_PROGRAM: A complete path to the executable that cleans the program, with all appropriate arguments. The executable must exist on every server where Oracle Grid Infrastructure is configured to run the application. If you do not specify a value for this attribute, then Oracle Clusterware uses an operating system-equivalent of the kill -9 command. For example:

```
/opt/my_app -clean
```

Note: The difference between STOP_PROGRAM and CLEAN_PROGRAM is that CLEAN_PROGRAM is a forced stop that stops an application ungracefully, and must always be able to stop an application or the application becomes unmanageable.

– PID_FILES: A comma-delimited list of complete paths to files that will be written by the application and contain a process ID (PID) to monitor. Failure of a single process is treated as a complete resource failure. For example:

```
/opt/app.pid
```

Note: The files that you specify in the PID_FILES attribute are read immediately after the START action completes and monitoring commences for the PIDs listed in the files.

– EXECUTABLE_NAMES: A comma-delimited list of names of executables that is created when the application starts and the state of these executables is subsequently monitored. Failure of a single executable is treated as a complete resource failure. For example:

```
my_app
```

> **Note:** You need specify only the complete name of the executables. This attribute does not accept the path of the executable or wild cards. The PIDs matching the executable names are cached immediately after the START action completes.

- CHECK_PROGRAMS: A list of complete paths to the executable that determines the state of the application. Reporting a non-running state by any of the applications is treated as a failure of the entire resource. For example:

  ```
  /opt/my_app -check
  ```

- ENVIRONMENT_FILE: A complete path to the file containing environment variables to source when starting the application. The file must be a text file containing *name*=*value* pairs, one per line. For example:

  ```
  /opt/my_app.env
  ```

- ENVIRONMENT_VARS: A comma-delimited list of *name*=*value* pairs to be included into the environment when starting an application. For example:

  ```
  USE_FILES=No, AUTO_START=Yes
  ```

- SEND_OUTPUT_ALWAYS: This attribute is responsible for sending the application output that is sent to STDOUT, which is then displayed. A value of 0 does not display any application output unless an action fails. When an action fails, whatever application output that has been saved by the agent is displayed. Any value greater than 0 displays every application output. The default value is 0. For example:

  ```
  SEND_OUTPUT_ALWAYS=1
  ```

> **Note:** If you do not specify the STOP_PROGRAM, CHECK_PROGRAMS, and CLEAN_PROGRAM attributes, then you must specify either PID_FILES or EXECUTABLE_NAMES, or Oracle Clusterware will not allow you to register a resource of this type.

If you specify all the attributes, then the following rules apply:

1. When stopping a resource, if you specified STOP_PROGRAM, then Oracle Clusterware calls STOP_PROGRAM. Otherwise, Oracle Clusterware uses an operating system-equivalent of the `kill -9` command on the PID obtained from either the PID_FILES or the EXECUTABLE_NAMES attribute.
2. When you need to establish the current state of an application, if you specified CHECK_PROGRAMS, then Oracle Clusterware calls CHECK_PROGRAMS. Otherwise, Oracle Clusterware uses an operating system-equivalent of the `ps -p` command with the PID obtained from either the PID_FILES or EXECUTABLE_NAMES attribute.
3. When cleaning a resource, if you specified CLEAN_PROGRAM, then Oracle Clusterware calls CLEAN_PROGRAM. Otherwise, Oracle Clusterware uses an operating system-equivalent of the `kill -9` command on the PID obtained from either the PID_FILES or the EXECUTABLE_NAMES attribute.

See Also:

- "`crsctl add resource`" on page E-6 for more information about and examples of using CRSCTL to add resources
- "Resource States" on page 9-12 for more information about resource state
- "Creating Resources that Use the generic_application Resource Type" on page 9-25

Agents

Oracle Clusterware manages applications when they are registered as resources with Oracle Clusterware. Oracle Clusterware has access to application-specific primitives that have the ability to start, stop, and monitor a specific resource. Oracle Clusterware runs all resource-specific commands through an entity called an *agent*.

An agent is a process that contains the agent framework and user code to manage resources. The *agent framework* is a library that enables you to plug in your application-specific code to manage customized applications. You program all of the actual application management functions, such as starting, stopping and checking the health of an application, into the agent. These functions are referred to as *entry points*.

The agent framework is responsible for invoking these entry point functions on behalf of Oracle Clusterware. Agent developers can use these entry points to plug in the required functionality for a specific resource regarding how to start, stop, and monitor a resource. Agents are capable of managing multiple resources.

Agent developers can set the following entry points as callbacks to their code:

- **ABORT**: If any of the other entry points hang, the agent framework calls the ABORT entry point to stop the ongoing action. If the agent developer does not supply a stop function, then the agent framework exits the agent program.
- **ACTION**: The ACTION entry point is Invoked when a custom action is invoked using the `clscrs_request_action` API of the `crsctl request action` command.
- **CHECK**: The CHECK (monitor) entry point acts to monitor the health of a resource. The agent framework periodically calls this entry point. If it notices any state change during this action, then the agent framework notifies Oracle Clusterware about the change in the state of the specific resource.
- **CLEAN**: The CLEAN entry point acts whenever there is a need to clean up a resource. It is a non-graceful operation that is invoked when users must forcefully terminate a resource. This command cleans up the resource-specific environment so that the resource can be restarted.
- **DELETE**: The DELETE entry point is invoked on every node where a resource can run when the resource is unregistered.
- **MODIFY**: The MODIFY entry point is invoked on every node where a resource can run when the resource profile is modified.
- **START**: The START entry point acts to bring a resource online. The agent framework calls this entry point whenever it receives the start command from Oracle Clusterware.
- **STOP**: The STOP entry points acts to gracefully bring down a resource. The agent framework calls this entry point whenever it receives the stop command from Oracle Clusterware.

START, STOP, CHECK, and CLEAN are mandatory entry points and the agent developer must provide these entry points when building an agent. Agent developers have several options to implement these entry points, including using C, C++, or scripts. It is also possible to develop agents that use both C or C++ and script-type entry points. When initializing the agent framework, if any of the mandatory entry points are not provided, then the agent framework invokes a script pointed to by the `ACTION_SCRIPT` resource attribute.

> **See Also:** "ACTION_SCRIPT" on page B-4 for information about this resource attribute

At any given time, the agent framework invokes only one entry point per application. If that entry point hangs, then the agent framework calls the ABORT entry point to end the current operation. The agent framework periodically invokes the CHECK entry point to determine the state of the resource. This entry point must return one of the following states as the resource state:

- `CLSAGFW_ONLINE`: The CHECK entry point returns ONLINE if the resource was brought up successfully and is currently in a functioning state. The agent framework continues to monitor the resource when it is in this state. This state has a numeric value of 0 for the `scriptagent`.
- `CLSAGFW_UNPLANNED_OFFLINE` and `CLSAGFW_PLANNED_OFFLINE`: The OFFLINE state indicates that the resource is not currently running. These two states have numeric values of 1 and 2, respectively, for the `scriptagent`.

  Two distinct categories exist to describe an resource's offline state: *planned* and *unplanned*.

  When the state of the resource transitions to OFFLINE through Oracle Clusterware, then it is assumed that the *intent* for this resource is to be offline (`TARGET=OFFLINE`), regardless of which value is returned from the CHECK entry point. However, when an agent detects that the state of a resource has changed independent of Oracle Clusterware (such as somebody stopping the resource through a non-Oracle interface), then the intent must be carried over from the agent to the Cluster Ready Services daemon (CRSD). The intent then becomes the determining factor for the following:

  – Whether to keep or to change the value of the resource's `TARGET` resource attribute. `PLANNED_OFFLINE` indicates that the `TARGET` resource attribute must be changed to OFFLINE *only if* the resource was running before. If the resource was not running (`STATE=OFFLINE`, `TARGET=OFFLINE`) and a request comes in to start it, then the value of the `TARGET` resource attribute changes to ONLINE. The start request then goes to the agent and the agent reports back to Oracle Clusterware a `PLANNED_OFFLINE` resource state, and the value of the `TARGET` resource attribute remains ONLINE. `UNPLANNED_OFFLINE` does not change the `TARGET` attribute.

  – Whether to leave the resource's state as `UNPLANNED_OFFLINE` or attempt to recover the resource by restarting it locally or failing it over to a another server in the cluster. The `PLANNED_OFFLINE` state makes CRSD leave the resource as is, whereas the `UNPLANNED_OFFLINE` state prompts resource recovery.
- `CLSAGFW_UNKNOWN`: The CHECK entry point returns UNKNOWN if the current state of the resource cannot be determined. In response to this state, Oracle Clusterware does not attempt to failover or to restart the resource. The agent framework continues to monitor the resource if the previous state of the resource was either ONLINE or PARTIAL. This state has a numeric value of 3 for the `scriptagent`.

- CLSAGFW_PARTIAL: The CHECK entry point returns PARTIAL when it knows that a resource is partially ONLINE and some of its services are available. Oracle Clusterware considers this state as partially ONLINE and does not attempt to failover or to restart the resource. The agent framework continues to monitor the resource in this state. This state has a numeric value of 4 for the scriptagent.
- CLSAGFW_FAILED: The CHECK entry point returns FAILED whenever it detects that a resource is not in a functioning state and some of its components have failed and some clean up is required to restart the resource. In response to this state, Oracle Clusterware calls the CLEAN action to clean up the resource. After the CLEAN action finishes, the state of the resource is expected to be OFFLINE. Next, depending on the policy of the resource, Oracle Clusterware may attempt to failover or restart the resource. Under no circumstances does the agent framework monitor failed resources. This state has a numeric value of 5 for the scriptagent.

The agent framework implicitly monitors resources in the states listed in Table 9–1 at regular intervals, as specified by the CHECK_INTERVAL or OFFLINE_CHECK_INTERVAL resource attributes.

> **See Also:** "CHECK_INTERVAL" on page B-6 and "OFFLINE_CHECK_INTERVAL" on page B-8 for more information about these resource attributes

Table 9–1 Agent Framework Monitoring Characteristics

State	Condition	Frequency
ONLINE	Always	CHECK_INTERVAL
PARTIAL	Always	CHECK_INTERVAL
OFFLINE	Only if the value of the OFFLINE_CHECK_INTERVAL resource attribute is greater than 0.	OFFLINE_CHECK_INTERVAL
UNKNOWN	Only monitored if the resource was previously being monitored because of any one of the previously mentioned conditions.	If the state becomes UNKNOWN after being ONLINE, then the value of CHECK_INTERVAL is used. Otherwise, there is no monitoring.

Whenever an agent starts, the state of all the resources it monitors is set to UNKNOWN. After receiving an initial probe request from Oracle Clusterware, the agent framework executes the CHECK entry point for all of the resources to determine their current states.

Once the CHECK action successfully completes for a resource, the state of the resource transitions to one of the previously mentioned states. The agent framework then starts resources based on commands issued from Oracle Clusterware. After the completion of every action, the agent framework invokes the CHECK action to determine the current resource state. If the resource is in one of the monitored states listed in Table 9–1, then the agent framework periodically executes the CHECK entry point to check for changes in resource state.

By default, the agent framework does not monitor resources that are offline. However, if the value of the OFFLINE_CHECK_INTERVAL attribute is greater than 0, then the agent framework monitors offline resources.

Built-in Agents

Oracle Clusterware uses agent programs (agents) to manage resources and includes the following built-in agents to protect applications:

- **appagent**: This agent (appagent.exe in Windows) automatically protects resources of the generic_application resource type and any resources in previous versions of Oracle Clusterware of the application resource type.

> **Note:** Oracle recommends that you not use the deprecated application resource type, which is only provided to support pre-Oracle Clusterware 11*g* release 2 (11.2) resources.

- **scriptagent**: Use this agent (scriptagent.exe in Windows) when using shell or batch scripts to protect an application. Both the cluster_resource and local_resource resource types are configured to use this agent, and any resources of these types automatically take advantage of this agent.

Additionally, you can create your own agents to manage your resources in any manner you want.

> **See Also:** "Resource Types" on page 9-2 for more information about resource types and "Building an Agent" on page 9-9 for more information about building custom agents

Action Scripts

An action script defines one or more actions to start, stop, check, or clean resources. The agent framework invokes these actions without the C/C++ actions. Using action scripts, you can build an agent that contains the C/C++ entry points and the script entry points. If all of the actions are defined in the action script, then you can use the script agent to invoke the actions defined in any action scripts.

Before invoking the action defined in the action script, the agent framework exports all the necessary attributes from the resource profile to the environment. Action scripts can log messages to the stdout/stderr, and the agent framework prints those messages in the agent logs. However, action scripts can use special tags to send the progress, warning, or error messages to the crs* client tools by prefixing one of the following tags to the messages printed to stdout/stderr:

```
CRS_WARNING:
CRS_ERROR:
CRS_PROGRESS:
```

The agent framework strips out the prefixed tag when it sends the final message to the crs* clients.

Resource attributes can be accessed from within an action script as environment variables prefixed with _CRS_. For example, the START_TIMEOUT attribute becomes an environment variable named _CRS_START_TIMEOUT.

> **See Also:**
>
> - "crsctl start resource" on page E-29 and "crsctl stop resource" on page E-34 for more information about using the -env option as a way to override attribute values for a single start or stop action, respectively
> - Example B–3 on page B-20 for an example of an action script for an agent

Building an Agent

Building an **agent** for a specific application involves the following steps:

1. Implement the **agent framework** entry points either in scripts, C, or C++.
2. Build the agent executable (for C and C++ agents).
3. Collect all the parameters needed by the entry points and define a new resource type. Set the AGENT_FILENAME attribute to the absolute path of the newly built executable.

Building and Deploying C and C++ Agents

Example C and C++ agents are included with Oracle Clusterware that demonstrate using the agent framework to implement high availability agents for applications. Appendix F describes an example of an agent called demoagent1.cpp. This agent manages a simple resource that represents a file on disk and performs the following tasks:

- On start: Creates the file
- On stop: Gracefully deletes the file
- On check: Detects whether the file is present
- On clean: Forcefully deletes the file

To describe this particular resource to Oracle Clusterware, you must first create a resource type that contains all the characteristic attributes for this resource class. In this case, the only attribute to be described is the name of the file to be managed. The following steps demonstrate how to set up the resource and its agent and test the functionality of the resource:

1. Compile the C++ agent using the demoagent1.cpp source file provided and a *makefile*. Modify the makefile based on the local compiler and linker paths and installation locations. The output is an executable named demoagent1. This example assumes that the executable is located in a directory named /path/to/ on every node of the cluster.
2. Use CRSCTL to add a new resource type, as follows:

   ```
   crsctl add type hotfile_type -basetype cluster_resource -attr
      "ATTRIBUTE=PATH_NAME,TYPE=string,DEFAULT_VALUE=default.txt,
      ATTRIBUTE=AGENT_FILENAME,TYPE=string,DEFAULT_VALUE=/path/to/demoagent1"
   ```

   In the preceding command example, PATH_NAME is the directory path for every resource of this type. Modify the value of PATH_NAME to the appropriate directory location on the disk.

   The AGENT_FILENAME attribute specifies the location of the agent binary that implements the resource management commands for this resource type. This step adds a new resource type to Oracle Clusterware.
3. Create a new resource based on the type that is defined in the previous step, as follows:

   ```
   $ crsctl add res file1 -type hotfile_type -attr "PATH_NAME=/var/log/file1.txt"
   $ crsctl add res file2 -type hotfile_type -attr "PATH_NAME=/var/log/file2.txt"
   ```

   The preceding commands add resources named file1 and file2 to be managed and monitored by Oracle Clusterware.
4. Start and stop the resources using CRSCTL, as follows:

```
$ crsctl start res file1
$ crsctl start res file2
$ crsctl relocate res file1
$ crsctl stop res file2
```

Oracle Clusterware creates and deletes the disk files as the resources are started and stopped.

> **See Also:** Appendix G, "Oracle Clusterware Agent Framework C Application Program Interfaces" for details of the agent framework, agent framework API reference, and supplied example agents

Registering a Resource in Oracle Clusterware

Register resources in Oracle Clusterware 12*c* using the `crsctl add resource` command.

> **Note:** The `CRS_REGISTER` and `CRS_PROFILE` commands are still available in the Oracle Clusterware home but are deprecated for this release.

To register an application as a resource:

```
$ crsctl add resource resource_name -type resource_type [-file file_path] | [-attr
"attribute_name='attribute_value', attribute_name='attribute_value', ..."]
```

Choose a name for the resource based on the application for which it is being created. For example, if you create a resource for an Apache Web server, then you might name the resource `myApache`.

The name of the resource type follows the `-type` option. You can specify resource attributes in either a text file specified with the `-file` option or in a comma-delimited list of resource attribute-value pairs enclosed in double quotation marks (`""`) following the `-attr` option. You must enclose space- or comma-delimited attribute names and values enclosed in parentheses in single quotation marks (`''`).

The following is an example of an attribute file:

```
PLACEMENT=favored
HOSTING_MEMBERS=node1 node2 node3
RESTART_ATTEMPTS@CARDINALITYID(1)=0
RESTART_ATTEMPTS@CARDINALITYID(2)=0
FAILURE_THRESHOLD@CARDINALITYID(1)=2
FAILURE_THRESHOLD@CARDINALITYID(2)=4
FAILURE_INTERVAL@CARDINALITYID(1)=300
FAILURE_INTERVAL@CARDINALITYID(2)=500
CHECK_INTERVAL=2
CARDINALITY=2
```

The following is an example of using the `-attr` option:

```
$ crsctl add resource resource_name -type resource_type [-attr
"PLACEMENT='favored', HOSTING_MEMBERS='node1 node2 node3', ..."]
```

See Also:

- "Adding User-Defined Resources" on page 9-22 for examples of using the `crsctl add resource` command
- "`crsctl add resource`" on page E-6 for more information about and examples of using the `crsctl add resource` command
- "Resource Attributes" on page B-1 for more information about resource attributes

Overview of Using Oracle Clusterware to Enable High Availability

Oracle Clusterware manages resources based on how you configure them to increase their availability. You can configure your resources so that Oracle Clusterware:

- Starts resources during cluster or server start
- Restarts resources when failures occur
- Relocates resources to other servers, if the servers are available

To manage your applications with Oracle Clusterware:

1. Use the `generic_application` resource type, write a custom script for the script agent, or develop a new agent.
2. Register your applications as resources with Oracle Clusterware.

   If a single application requires that you register multiple resources, you may be required to define relevant dependencies between the resources.
3. Assign the appropriate privileges to the resource.
4. Start or stop your resources.

When a resource fails, Oracle Clusterware attempts to restart the resource based on attribute values that you provide when you register an application or process as a resource. If a server in a cluster fails, then you can configure your resources so that processes that were assigned to run on the failed server restart on another server. Based on various resource attributes, Oracle Clusterware supports a variety of configurable scenarios.

When you register a resource in Oracle Clusterware, the relevant information about the application and the resource-relevant information, is stored in the Oracle Cluster Registry (OCR). This information includes:

- **Path to the action script or application-specific agent**: This is the absolute path to the script or application-specific agent that defines the start, stop, check, and clean actions that Oracle Clusterware performs on the application.

  See Also: "Agents" on page 9-5 for more information about these actions

- **Privileges**: Oracle Clusterware has the necessary privileges to control all of the components of your application for high availability operations, including the right to start processes that are owned by other user identities. Oracle Clusterware must run as a privileged user to control applications with the correct start and stop processes.
- **Resource Dependencies**: You can create relationships among resources that imply an operational ordering or that affect the placement of resources on servers in the cluster. For example, Oracle Clusterware can only start a resource that has a hard

start dependency on another resource if the other resource is running. Oracle Clusterware prevents stopping a resource if other resources that depend on it are running. However, you can force a resource to stop using the `crsctl stop resource -f` command, which first stops all resources that depend on the resource being stopped.

This section includes the following topics:

- Resource Attributes
- Resource States
- Resource Dependencies
- Resource Placement

Resource Attributes

Resource attributes define how Oracle Clusterware manages resources of a specific resource type. Each resource type has a unique set of attributes. Some resource attributes are specified when you register resources, while others are internally managed by Oracle Clusterware.

Note: Where you can define new resource attributes, you can only use US-7 ASCII characters.

See Also: Appendix B, "Oracle Clusterware Resource Reference" for complete details of resource attributes

Resource States

Every resource in a cluster is in a particular state at any time. Certain actions or events can cause that state to change.

Table 9–2 lists and describes the possible **resource states**.

Table 9–2 Possible Resource States

State	Description
ONLINE	The resource is running.
OFFLINE	The resource is not running.
UNKNOWN	An attempt to stop the resource has failed. Oracle Clusterware does not actively monitor resources that are in this state. You must perform an application-specific action to ensure that the resource is offline, such as stop a process, and then run the `crsctl stop resource` command to reset the state of the resource to OFFLINE.

Table 9–2 (Cont.) Possible Resource States

State	Description
INTERMEDIATE	A resource can be in the INTERMEDIATE state because of one of two events: 1. Oracle Clusterware cannot determine the state of the resource but the resource was either attempting to go online or was online the last time its state was precisely known. Usually, the resource transitions out of this state on its own over time, as the conditions that impeded the check action no longer apply. 2. A resource is partially online. For example, the Oracle Database VIP resource fails over to another server when its home server leaves the cluster. However, applications cannot use this VIP to access the database while it is on a non-home server. Similarly, when an Oracle Database instance is started and not open, the resource is partially online: it is running but is not available to provide services. Oracle Clusterware actively monitors resources that are in the INTERMEDIATE state and, typically, you are not required to intervene. If the resource is in the INTERMEDIATE state due to the preceding reason 1, then as soon as the state of the resource is established, Oracle Clusterware transitions the resource out of the INTERMEDIATE state. If the resource is in the INTERMEDIATE state due to the preceding reason 2, then it stays in this state if it remains partially online. For example, the home server of the VIP must rejoin the cluster so the VIP can switch over to it. A database administrator must issue a command to open the database instance. In either case, however, Oracle Clusterware transitions the resource out of the INTERMEDIATE state automatically as soon as it is appropriate.Use the `STATE_DETAILS` resource attribute to explain the reason for a resource being in the INTERMEDIATE state and provide a solution to transition the resource out of this state.

Resource Dependencies

You can configure resources to be dependent on other resources, so that the dependent resources can only start or stop when certain conditions of the resources on which they depend are met. For example, when Oracle Clusterware attempts to start a resource, it is necessary for any resources on which the initial resource depends to be running and in the same location. If Oracle Clusterware cannot bring the resources online, then the initial (dependent) resource cannot be brought online, either. If Oracle Clusterware stops a resource or a resource fails, then any dependent resource is also stopped.

Some resources require more time to start than others. Some resources must start whenever a server starts, while other resources require a manual start action. These and many other examples of resource-specific behavior imply that each resource must be described in terms of how it is expected to behave and how it relates to other resources (resource dependencies).

You can configure resources so that they depend on Oracle resources. When creating resources, however, do not use an *ora* prefix in the resource name. This prefix is reserved for Oracle use only.

Previous versions of Oracle Clusterware included only two dependency specifications: the `REQUIRED_RESOURCES` resource attribute and the `OPTIONAL_RESOURCES` resource attribute. The REQUIRED_RESOURCES resource attribute applied to both start and stop resource dependencies.

Note: The REQUIRED_RESOURCES and OPTIONAL_RESOURCES resource attributes are still available only for resources of application type. Their use to define resource dependencies in Oracle Clusterware 12*c* is deprecated.

Resource dependencies are separated into start and stop categories. This separation improves and expands the start and stop dependencies between resources and resource types.

This section includes the following topics:

- Start Dependencies
- Stop Dependencies

Start Dependencies

Oracle Clusterware considers start dependencies contained in the profile of a resource when the **start effort evaluation** for that resource begins. You specify start dependencies for resources using the START_DEPENDENCIES resource attribute. You can use modifiers on each dependency to further configure the dependency.

See Also: "START_DEPENDENCIES" on page B-11 for more information about the resource attribute, modifiers, and usage

This section includes descriptions of the following START dependencies:

- attraction
- dispersion
- exclusion
- hard
- pullup
- weak

attraction

If resource A has an attraction dependency on resource B, then Oracle Clusterware prefers to place resource A on servers hosting resource B. Dependent resources, such as resource A in this case, are more likely to run on servers on which resources to which they have attraction dependencies are running. Oracle Clusterware places dependent resources on servers with resources to which they are attracted.

You can configure the attraction start dependency with the following constraints:

- START_DEPENDENCIES=attraction(intermediate:resourceB)

  Use the intermediate modifier to specify whether the resource is attracted to resources that are in the INTERMEDIATE state.

- START_DEPENDENCIES=attraction(type:resourceB.type)

  Use the type modifier to specify whether the dependency acts on a particular resource type. The dependent resource is attracted to the server hosting the greatest number of resources of a particular type.

Note: Previous versions of Oracle Clusterware used the now deprecated `OPTIONAL_RESOURCES` attribute to express attraction dependency.

dispersion

If you specify the `dispersion` start dependency for a resource, then Oracle Clusterware starts this resource on a server that has the fewest number of resources to which this resource has dispersion. Resources with dispersion may still end up running on the same server if there are not enough servers to which to disperse them.

You can configure the `dispersion` start dependency with the following modifiers:

- `START_DEPENDENCIES=dispersion(intermediate:resourceB)`

  Use the `intermediate` modifier to specify that Oracle Clusterware disperses resource A whether resource B is either in the `ONLINE` or `INTERMEDIATE` state.

- `START_DEPENDENCIES=dispersion:active(resourceB)`

  Typically, dispersion is only applied when starting resources. If at the time of starting, resources that disperse each other start on the same server (because there are not enough servers at the time the resources start), then Oracle Clusterware leaves the resources alone once they are running, even when more servers join the cluster. If you specify the `active` modifier, then Oracle Clusterware reapplies dispersion on resources later when new servers join the cluster.

- `START_DEPENDENCIES=dispersion(pool:resourceB)`

  Use the `pool` modifier to specify that Oracle Clusterware disperses the resource to a different server pool rather than to a different server.

exclusion

The `exclusion` start dependency contains a clause that defines the exclusive relationship between resources while starting. Resources that have the `exclusion` start dependency cannot run on the same node. For example, if resource A has an `exclusion` start dependency on resource B, then the CRSD policy provides the following options when resource B is already running on the server where resource A needs to start:

- Deny the start of resource A if resource B is already running.
- Start resource A by preempting resource B. There are two variations to the preempt operation:
  - Resource B is stopped and, if possible, restarted on another node. Resource A is subsequently started.
  - Resource A is started first. Subsequently, resource B is stopped and, if possible, restarted on another node.

You can configure the `exclusion` start dependency with the following modifiers:

- `START_DEPENDENCIES=exclusion([[preempt_pre: | preempt_post:]]` *`target_resource_name`* `| type:`*`target_resource_type`*`]*)`

  All modifiers specified are per resource or resource type. Oracle Clusterware permits only one `exclusion` dependency per resource dependency tree. Without any `preempt` modifier, CRSD will only attempt to start the resource if all of its target resources are offline.

- preempt_pre: If you choose this preempt modifier, then CRSD stops the specified target resource or resources defined by a specific resource type before starting the source resource. If restarting the stopped resources is possible, then CRSD can do this concurrently while starting the preempting resource.
- preempt_post: If you choose this preempt modifier, then, after starting the source resource, CRSD stops and relocates, if possible, the specified target resource or resources defined by a specific resource type.

If CRSD cannot stop the target resources successfully, or cannot start the source resource, then the entire operation fails. Oracle Clusterware then attempts to return the affected resources to their original state, if possible.

hard

Define a hard start dependency for a resource if another resource must be running before the dependent resource can start. For example, if resource A has a hard start dependency on resource B, then resource B must be running before resource A can start.

Note: Oracle recommends that resources with hard start dependencies also have pullup start dependencies.

You can configure the hard start dependency with the following constraints:

- START_DEPENDENCIES=hard(global:resourceB)

  By default, resources A and B must be located on the same server (collocated). Use the global modifier to specify that resources need not be collocated. For example, if resource A has a hard(global:resourceB) start dependency on resource B, then, if resource B is running on any node in the cluster, resource A can start.

- START_DEPENDENCIES=hard(intermediate:resourceB)

  Use the intermediate modifier to specify that the dependent resource can start if a resource on which it depends is in either the ONLINE or INTERMEDIATE state.

- START_DEPENDENCIES=hard(type:resourceB.type)

  Use the type modifier to specify whether the hard start dependency acts on a particular resource or a resource type. For example, if you specify that resource A has a hard start dependency on the resourceB.type type, then if any resource of the resourceB.type type is running, resource A can start.

- START_DEPENDENCIES=hard(uniform:resourceB)

  Use the uniform modifier to attempt to start all instances of resource B, but only one instance, at least must start to satisfy the dependency.

- START_DEPENDENCIES=hard(resourceB, intermediate:resourceC, intermediate:global:type:resourceC.type)

  You can combine modifiers and specify multiple resources in the START_DEPENDENCIES resource attribute.

Note: Separate modifier clauses with commas. The type modifier clause must always be the last modifier clause in the list and the type modifier must always directly precede the type.

pullup

Use the pullup start dependency if resource A must automatically start whenever resource B starts. This dependency only affects resource A if it is not running. As is the case for other dependencies, pullup may cause the dependent resource to start on any server. Use the pullup dependency whenever there is a hard stop dependency, so that if resource A depends on resource B and resource B fails and then recovers, then resource A is restarted.

> **Note:** Oracle recommends that resources with hard start dependencies also have pullup start dependencies.

You can configure the pullup start dependency with the following constraints:

- START_DEPENDENCIES=pullup(intermediate:resourceB)

  Use the intermediate modifier to specify whether resource B can be either in the ONLINE or INTERMEDIATE state to start resource A.

  If resource A has a pullup dependency on multiple resources, then resource A starts only when all resources upon which it depends, start.

- START_DEPENDENCIES=pullup:always(resourceB)

  Use the always modifier to specify whether Oracle Clusterware starts resource A despite the value of its TARGET attribute, whether it is ONLINE or OFFLINE. By default, without using the always modifier, pullup only starts resources if the value of the TARGET attribute of the dependent resource is ONLINE.

- START_DEPENDENCIES=pullup(type:resourceB.type)

  Use the type modifier to specify that the dependency acts on a particular resource type.

weak

If resource A has a weak start dependency on resource B, then an attempt to start resource A attempts to start resource B, if resource B is not running. The result of the attempt to start resource B is, however, of no consequence to the result of starting resource A.

You can configure the weak start dependency with the following constraints:

- START_DEPENDENCIES=weak(global:resourceB)

  By default, resources A and B must be collocated. Use the global modifier to specify that resources need not be collocated. For example, if resource A has a weak(global:resourceB) start dependency on resource B, then, if resource B is running on any node in the cluster, resource A can start.

- START_DEPENDENCIES=weak(concurrent:resourceB)

  Use the concurrent modifier to specify that resource A and resource B can start concurrently, instead of waiting for resource B to start, first.

- START_DEPENDENCIES=weak(type:resourceB.type)

  Use the type modifier to specify that the dependency acts on a resource of a particular resource type, such as resourceB.type.

- START_DEPENDENCIES=weak(uniform:resourceB)

  Use the uniform modifier to attempt to start all instances of resource B.

Stop Dependencies

Oracle Clusterware considers stop dependencies between resources whenever a resource is stopped (the resource state changes from `ONLINE` to any other state).

hard

If resource A has a `hard` stop dependency on resource B, then resource A must be stopped when B stops running. The two resources may attempt to start or relocate to another server, depending upon how they are configured. Oracle recommends that resources with `hard` stop dependencies also have `hard` start dependencies.

You can configure the `hard` stop dependency with the following modifiers:

- `STOP_DEPENDENCIES=hard(intermedite:resourceB)`

  Use the `intermediate` modifier to specify whether resource B must be in either the `ONLINE` or `INTERMEDIATE` state for resource A to stay online.

- `STOP_DEPENDENCIES=hard(global:resourceB)`

  Use the `global` modifier to specify whether resource A requires that resource B be present on the same server or on any server in the cluster to remain online. If this constraint is not specified, then resources A and B must be running on the same server. Oracle Clusterware stops resource A when that condition is no longer met.

- `STOP_DEPENDENCIES=hard(shutdown:resourceB)`

  Use the `shutdown` modifier to stop the resource only when you shut down the Oracle Clusterware stack using either the `crsctl stop crs` or `crsctl stop cluster` commands.

> **See Also:** "STOP_DEPENDENCIES" on page B-14 for more information about modifiers

Affect of Resource Dependencies on Resource State Recovery

When a resource goes from a running to a non-running state, while the intent to have it running remains unchanged, this transition is called a *resource failure*. At this point, Oracle Clusterware applies a resource state recovery procedure that may try to restart the resource locally, relocate it to another server, or just stop the dependent resources, depending on the high availability policy for resources and the state of entities at the time.

When two or more resources depend on each other, a failure of one of them may end up causing the other to fail, as well. In most cases, it is difficult to control or even predict the order in which these failures are detected. For example, even if resource A depends on resource B, Oracle Clusterware may detect the failure of resource B after the failure of resource A.

This lack of failure order predictability can cause Oracle Clusterware to attempt to restart dependent resources in parallel, which, ultimately, leads to the failure to restart some resources, because the resources upon which they depend are being restarted out of order.

In this case, Oracle Clusterware reattempts to restart the dependent resources locally if either or both the `hard` stop and `pullup` dependencies are used. For example, if resource A has either a `hard` stop dependency or `pullup` dependency, or both, on resource B, and resource A fails because resource B failed, then Oracle Clusterware may end up trying to restart both resources at the same time. If the attempt to restart resource A fails, then as soon as resource B successfully restarts, Oracle Clusterware reattempts to restart resource A.

Resource Placement

As part of the start effort evaluation, the first decision that Oracle Clusterware must make is where to start (or place) the resource. Making such a decision is easy when the caller specifies the target server by name. If a target server is not specified, however, then Oracle Clusterware attempts to locate the best possible server for placement given the resource's configuration and the current state of the cluster.

Oracle Clusterware considers a resource's placement policy first and filters out servers that do not fit with that policy. Oracle Clusterware sorts the remaining servers in a particular order depending on the value of the `PLACEMENT` resource attribute of the resource.

> **See Also:** "Application Placement Policies" on page 9-28 for more information about the `PLACEMENT` resource attribute

The result of this consideration is a maximum of two lists of candidate servers on which Oracle Clusterware can start the resource. One list contains preferred servers and the other contains possible servers. The list of preferred servers will be empty if the value of the `PLACEMENT` resource attribute for the resource is set to `balanced` or `restricted`. The placement policy of the resource determines on which server the resource wants to run. Oracle Clusterware considers preferred servers over possible servers, if there are servers in the preferred list.

Oracle Clusterware then considers the resource's dependencies to determine where to place the resource, if any exist. The `attraction` and `dispersion` start dependencies affect the resource placement decision, as do some dependency modifiers. Oracle Clusterware applies these placement hints to further order the servers in the two previously mentioned lists. Note that Oracle Clusterware processes each list of servers independently, so that the effect of the resource's placement policy is not confused by that of dependencies.

Finally, Oracle Clusterware chooses the first server from the list of preferred servers, if any servers are listed. If there are no servers on the list of preferred servers, then Oracle Clusterware chooses the first server from the list of possible servers, if any servers are listed. When no servers exist in either list, Oracle Clusterware generates a resource placement error.

> **Note:** Neither the placement policies nor the dependencies of the resources related to the resource Oracle Clusterware is attempting to start affect the placement decision.

Registering an Application as a Resource

This section presents examples of the procedures for registering an application as a resource in Oracle Clusterware. The procedures instruct you how to add an Apache Web server as a resource to Oracle Clusterware.

The examples in this section assume that the Oracle Clusterware administrator has full administrative privileges over Oracle Clusterware and the user or group that owns the application that Oracle Clusterware is going to manage. Once the registration process is complete, Oracle Clusterware can start any application on behalf of any operating system user.

Oracle Clusterware distinguishes between an owner of a registered resource and a user. The owner of a resource is the operating system user under which the agent runs.

The ACL resource attribute of the resource defines permissions for the users and the owner. Only root can modify any resource.

See Also: "Role-Separated Management" on page 2-1

Notes:

- Oracle Clusterware commands prefixed with crs_ are deprecated with this release. CRSCTL commands replace those commands. See Appendix E, "Oracle Clusterware Control (CRSCTL) Utility Reference" for a list of CRSCTL commands and their corresponding crs_ commands.
- Do not use CRSCTL commands on any resources that have names prefixed with ora (because these are Oracle resources), unless My Oracle Support directs you to do so.

  To configure Oracle resources, use the server control utility, SRVCTL, which provides you with all configurable options.

This section includes the following topics:

- Creating an Application VIP Managed by Oracle Clusterware
- Adding User-Defined Resources
- Changing Resource Permissions
- Application Placement Policies
- Unregistering Applications and Application Resources

Creating an Application VIP Managed by Oracle Clusterware

If clients of an application access the application through a network, and the placement policy for the application allows it to fail over to another node, then you must register a virtual internet protocol address (VIP) on which the application depends. An application VIP is a cluster resource that Oracle Clusterware manages (Oracle Clusterware provides a standard VIP agent for application VIPs). You should base any new application VIPs on this VIP type to ensure that your system experiences consistent behavior among all of the VIPs that you deploy in your cluster.

While you can add a VIP in the same way that you can add any other resource that Oracle Clusterware manages, Oracle recommends using the *Grid_home*/bin/appvipcfg command-line utility to create or delete an application VIP on the default network for which the ora.net1.network resource is created by default.

To create an application VIP, use the following syntax:

```
appvipcfg create -network=network_nummber -ip=ip_address -vipname=vip_name
-user=user_name [-group=group_name] [-failback=0 | 1]
```

Note: You can modify the VIP name while the resource remains online, without restarting the resource.

When you create an application VIP on a default network, set -network=1.

To create an application VIP on a non-default network, you may have to first create the network using the `srvctl add network` command. Then you can create the application VIP, setting `-network=`*`non-default_network_number`*.

In an Oracle Flex Cluster, you can also add a Leaf Node network resource for application VIPs, so that applications can run on Leaf Nodes using the `srvctl add network` command, use the following syntax:

```
srvctl add network -netnum=network_number -subnet subnet/netmask[/if1[|if2|...]]
-leaf
```

See Also:

- *Oracle Real Application Clusters Administration and Deployment Guide* for more information about the `srvctl add network` command
- Chapter 4, "Oracle Flex Clusters" for more information about Leaf Nodes

To delete an application VIP, use the following syntax:

```
appvipcfg delete -vipname=vip_name
```

In the preceding syntax examples, *`network_number`* is the number of the network, *`ip_address`* is the IP address, *`vip_name`* is the name of the VIP, *`user_name`* is the name of the user who installed Oracle Database, and *`group_name`* is the name of the group. The default value of the `-failback` option is 0. If you set the option to 1, then the VIP (and therefore any resources that depend on VIP) fails back to the original node when it becomes available again.

For example, as `root`, run the following command:

```
# Grid_home/bin/appvipcfg create -network=1 -ip=148.87.58.196 -vipname=appsVIP
-user=root
```

The script only requires a network number, the IP address, and a name for the VIP resource, in addition to the user that owns the application VIP resource. A VIP resource is typically owned by `root` because VIP related operations require root privileges.

To delete an application VIP, use the same script with the `delete` option. This option accepts the VIP name as a parameter. For example:

```
# Grid_home/bin/appvipcfg delete -vipname=appsVIP
```

After you have created the application VIP using this configuration script, you can view the VIP profile using the following command:

```
Grid_home/bin/crsctl status res appsVIP -p
```

Verify and, if required, modify the following parameters using the *`Grid_home`*`/bin/crsctl modify res` command.

See Also: Appendix B, "Oracle Clusterware Resource Reference" for detailed information about using CRSCTL commands

The `appvipcfg` script requires that you specify the `-network` option, even if `-network=1`.

As the Oracle Database installation owner, start the VIP resource:

```
$ crsctl start resource appsVIP
```

Adding an Application VIP with Oracle Enterprise Manager

To add an application VIP with Oracle Enterprise Manager:

1. Log into Oracle Enterprise Manager Cloud Control.
2. Select the cluster target that you want to modify.
3. From the cluster target menu, select **Administration** > **Resources** > **Manage**.
4. Enter a cluster administrator user name and password to display the Manage Resources page.
5. Click **Add Application VIP**.
6. Enter a name for the VIP in the **Name** field.
7. Enter a network number in the **Network Number** field.
8. Enter an IP address for the VIP in the **Internet Protocol Address** field.
9. Enter **root** in the **Primary User** field. Oracle Enterprise Manager defaults to whatever user name you are logged in as.
10. Select **Start the resource after creation** if you want the VIP to start immediately.
11. Click **Continue** to display the Confirmation: Add VIP Resource page.
12. Enter **root** and the root password as the cluster credentials.
13. Click **Continue** to create the application VIP.

Adding User-Defined Resources

You can add resources to Oracle Clusterware at any time. However, if you add a resource that depends on another resource, then you must first add the resource upon which it is dependent.

In the examples in this section, assume that an action script, `myApache.scr`, resides in the `/opt/cluster/scripts` directory on each node to facilitate adding the resource to the cluster. Assume also that a server pool has been created to host an application. This server pool is not a sub-pool of Generic, but instead it is used to host the application in a top-level server pool.

> **See Also:** "Examples of Action Scripts for Third-party Applications" on page B-18 to see an example of an action script

> **Note:** Oracle recommends that you use shared storage, such as Oracle Automatic Storage Management Cluster File System (Oracle ACFS), to store action scripts to decrease script maintenance.

This section includes the following topics:

- Deciding on a Deployment Scheme
- Adding a Resource to a Specified Server Pool
- Adding a Resource Using a Server-Specific Deployment
- Creating Resources that Use the generic_application Resource Type

Deciding on a Deployment Scheme

You must decide whether to use administrator or policy management for the application. Use administrator management for smaller, two-node configurations, where your cluster configuration is not likely to change. Use policy management for more dynamic configurations when your cluster consists of more than two nodes. For example, if a resource only runs on node 1 and node 2 because only those nodes have the necessary files, then administrator management is probably more appropriate.

Oracle Clusterware supports the deployment of applications in access-controlled server pools made up of anonymous servers and strictly based on the desired pool size. Cluster policies defined by the administrator can and must be used in this case to govern the server assignment with desired sizes and levels of importance. Alternatively, a strict or preferred server assignment can be used, in which resources run on specifically named servers. This represents the pre-existing model available in earlier releases of Oracle Clusterware now known as administrator management.

Conceptually, a cluster hosting applications developed and deployed in both of the deployment schemes can be viewed as two logically separated groups of servers. One server group is used for server pools, enabling role separation and server capacity control. The other server group assumes a fixed assignment based on named servers in the cluster.

To manage an application using either deployment scheme, you must create a server pool before adding the resource to the cluster. A built-in server pool named *Generic* always owns the servers used by applications of administrator-based management. The Generic server pool is a logical division and can be used to separate the two parts of the cluster using different management schemes.

For third party developers to use the model to deploy applications, server pools must be used. To take advantage of the pre-existing application development and deployment model based on named servers, sub-pools of Generic (server pools that have Generic as their parent pool, defined by the server pool attribute `PARENT_POOLS`) must be used. By creating sub-pools that use Generic as their parent and enumerating servers by name in the sub-pool definitions, applications ensure that named servers are in Generic and are used exclusively for applications using the named servers model.

Adding a Resource to a Specified Server Pool

Use the `crsctl add resource` command to add a resource to a server pool.

To add the Apache Web server to a specific server pool as a resource using the policy-based deployment scheme, run the following command as the user that is supposed to run the Apache Server (for an Apache Server this is typically the `root` user):

```
$ crsctl add resource myApache -type cluster_resource -attr
  "ACTION_SCRIPT=/opt/cluster/scripts/myapache.scr,
   PLACEMENT=restricted,
   SERVER_POOLS=server_pool_list,
   CHECK_INTERVAL=30,
   RESTART_ATTEMPTS=2,
   START_DEPENDENCIES=hard(appsvip),
   STOP_DEPENDENCIES=hard(appsvip)"
```

> **See Also:** "Examples of Action Scripts for Third-party Applications" on page B-18

In the preceding example, `myApache` is the name of the resource added to the cluster.

Notes:

- You must enclose comma or space-delimited attribute values in single quotation marks (' ') to avoid errors. If you enclose single attributes values in single quotation marks, they are ignored and no errors ensue.
- A resource name cannot begin with a period nor with the character string *ora*.

The resource is configured, as follows:

- The resource is a `cluster_resource` type.
- `ACTION_SCRIPT=/opt/cluster/scripts/myapache.scr`: The path to the required action script.
- `PLACEMENT=restricted`

  See Also: "Application Placement Policies" on page 9-28 for more information about the `PLACEMENT` resource attribute

- `SERVER_POOLS=`*`server_pool_list`*: This resource can only run in the server pools specified in a space-delimited list.
- `CHECK_INTERVAL=30`: Oracle Clusterware checks this resource every 30 seconds to determine its status.
- `RESTART_ATTEMPTS=2`: Oracle Clusterware attempts to restart this resource twice before failing it over to another node.
- `START_DEPENDENCIES=hard(appsvip)`: This resource has a hard START dependency on the appsvip resource. The appsvip resource must be online in order for myApache to start.
- `STOP_DEPENDENCIES=hard(appsvip)`: This resource has a hard STOP dependency on the appsvip resource. The myApache resource stops if the appsvip resource goes offline.

Adding a Resource Using a Server-Specific Deployment

To add the Apache Web server as a resource that uses a named server deployment, assume that you add the resource to a server pool that is, by definition, a sub-pool of the Generic server pool. You create server pools that are sub-pools of Generic using the `crsctl add serverpool` command. These server pools define the Generic server pool as their parent in the server pool attribute `PARENT_POOLS`. In addition, they include a list of server names in the `SERVER_NAMES` parameter to specify the servers that should be assigned to the respective pool. For example:

```
$ crsctl add serverpool myApache_sp -attr
  "PARENT_POOLS=Generic, SERVER_NAMES=host36 host37"
```

After you create the sub-pool, add the Apache Web server resource, as follows:

```
$ crsctl add resource myApache -type cluster_resource -attr
  "ACTION_SCRIPT=/opt/cluster/scripts/myapache.scr,
   PLACEMENT='restricted',
   SERVER_POOLS=myApache_sp,
   CHECK_INTERVAL='30',
   RESTART_ATTEMPTS='2',
   START_DEPENDENCIES='hard(appsvip)',
   STOP_DEPENDENCIES='hard(appsvip)'"
```

Note: A resource name cannot begin with a period nor with the character string *ora*.

In addition, note that when adding a resource using a server-specific deployment, the server pools listed in the `SERVER_POOLS` resource parameter must be sub-pools under Generic.

See Also: Chapter 3, "Policy-Based Cluster and Capacity Management"

Creating Resources that Use the generic_application Resource Type

Use the `crsctl add resource` command to create resources using the `generic_application` resource type to model any type of application requiring high availability without having to create any action scripts. This section includes two examples for Linux/UNIX platforms of creating resources that use the `generic_application` resource type.

See Also: "Resource Types" on page 9-2 for more information about the `generic_application` resource type

In the following command example, a Samba server resource is created for high availability:

```
$ crsctl add resource samba1 -type generic_application -attr
  "START_PROGRAM='/etc/init.d/smb start',
   STOP_PROGRAM='/etc/init.d/smb stop',
   CLEAN_PROGRAM='/etc/init.d/smb stop',
   PID_FILES='/var/run/smbd.pid,/var/run/nmbd.pid'"
```

In the preceding example, the attributes that define the resource are configured, as follows:

- `START_PROGRAM='/etc/init.d/smb start'`: This attribute contains the complete path and arguments to the script that starts the Samba server
- `STOP_PROGRAM='/etc/init.d/smb stop'`: This attribute contains the complete path and arguments to the script that stops the Samba server
- `CLEAN_PROGRAM='/etc/init.d/smb stop'`: This attribute contains the complete path and arguments to the script that forcefully terminates and cleans up the Samba server in case there is any failure in starting or stopping the server
- `PID_FILES='/var/run/smbd.pid,/var/run/nmbd.pid'`: This attribute contains the paths to the text files listing the process IDs (PIDs) that must be monitored to ensure that the Samba server and all its components are running

Notes:

- If script-based monitoring is required for this Samba server configuration, then you can use the CHECK_PROGRAMS attribute instead of the PID_FILES attribute, as follows:

  ```
  CHECK_PROGRAMS='/etc/init.d/smb status'
  ```

- You can specify standard Oracle Clusterware placement and cardinality properties by configuring the HOSTING_MEMBERS, SERVER_POOLS, PLACEMENT, and CARDINALITY attributes of the Samba server resource.

In the second command example, a database file server (DBFS) resource is created for high availability. The DBFS provides a Filesystem in Userspace (FUSE) file system to access data stored in an Oracle Database.

See Also: *Oracle Database SecureFiles and Large Objects Developer's Guide* for more information about DBFS

You can use the generic_application resource type to define a resource that corresponds to the DBFS file system. You can use this DBFS resource to start, stop, monitor, and failover the DBFS file system mount point. The command syntax to create this resource is as follows:

```
$ crsctl add resource dbfs1 -type generic_application -attr
  "START_PROGRAM='/app/oracle/12.1/bin/dbfs_client -o wallet
   /@inst1 /scratch/mjkeenan/data/dbfs_mount',
   STOP_PROGRAM='/bin/fusermount -u /scratch/mjkeenan/data/dbfs_mount',
   CHECK_PROGRAMS='ls /scratch/mjkeenan/data/dbfs_mount/dbfsdata1',
   ENVIRONMENT_VARS='ORACLE_HOME=/app/oracle/12.1,
         LD_LIBRARY_PATH=/app/oracle/12.1/lib:/app/oracle/12.1/rdbms/lib,
         TNS_ADMIN=/app/oracle/12.1/network/admin',
   CLEAN_PROGRAM='/bin/fusermount -u -z /scratch/mjkeenan/data/dbfs_mount',
   START_DEPENDENCIES='hard(ora.inst1_srv.svc)',
   STOP_DEPENDENCIES='hard(ora.inst1_srv.svc)'"
```

In addition to the mandatory START_PROGRAM, STOP_PROGRAM, CHECK_PROGRAMS, and CLEAN_PROGRAM attributes, the above example also includes the following:

- The ENVIRONMENT_VARS attribute specifies custom environment variables that are passed when starting or stopping the program
- The START_DEPENDENCIES and STOP_DEPENDENCIES dependency attributes create a start and stop dependency on the database service that is the underlying database store of the DBFS file system

  You can create dependencies on to the DBFS resource for higher-level application resources based on the application requirements of the DBFS file system.

Notes:

- The ORACLE_HOME directory shown in the preceding syntax is an example.
- You can specify standard Oracle Clusterware placement and cardinality properties by configuring the HOSTING_MEMBERS, SERVER_POOLS, PLACEMENT, and CARDINALITY attributes of the DBFS file system resource.

Adding Resources Using Oracle Enterprise Manager

To add resources to Oracle Clusterware using Oracle Enterprise Manager:

1. Log into Oracle Enterprise Manager Cloud Control.
2. Select the cluster target that you want to modify.
3. From the cluster target menu, select **Administration** > **Resources** > **Manage**.
4. Enter a cluster administrator user name and password to display the Add Resource page.
5. Enter a name for the resource in the **Name** field.

   Note: A resource name cannot begin with a period nor with the character string *ora*.

6. Choose either **cluster_resource** or **local_resource** from the **Resource Type** drop down.
7. Optionally, enter a description of the resource in the **Description** field.
8. Select **Start the resource after creation** if you want the resource to start immediately.
9. The optional parameters in the **Placement** section define where in a cluster Oracle Clusterware places the resource.

   See Also: "Application Placement Policies" on page 9-28 for more information about placement

   The attributes in this section correspond to the attributes described in Appendix B, "Oracle Clusterware Resource Reference".

10. In the **Action Program** section, choose from the **Action Program** drop down whether Oracle Clusterware calls an action script, an agent file, or both to manage the resource.

    You must also specify a path to the script, file, or both, depending on what you select from the drop down.

    If you choose **Action Script**, then you can click **Create New Action Script** to use the Oracle Enterprise Manager action script template to create an action script for your resource, if you have not yet done so.

11. To further configure the resource, click **Attributes**. On this page, you can configure start, stop, and status attributes, and offline monitoring and any attributes that you define.

12. Click **Advanced Settings** to enable more detailed resource attribute configurations.
13. Click **Dependencies** to configure start and stop dependencies between resources.

    > **See Also:** "Resource Dependencies" on page 9-13 for more information about dependencies

14. Click **Submit** when you finish configuring the resource.

Changing Resource Permissions

Oracle Clusterware manages resources based on the permissions of the user who added the resource. The user who first added the resource owns the resource and the resource runs as the resource owner. Certain resources must be managed as `root`. If a user other than `root` adds a resource that must be run as `root`, then the permissions must be changed as `root` so that `root` manages the resource, as follows:

1. Change the permission of the named resource to `root` by running the following command as `root`:

   ```
   # crsctl setperm resource resource_name -o root
   ```

2. As the user who installed Oracle Clusterware, enable the Oracle Database installation owner (`oracle`, in the following example) to run the script, as follows:

   ```
   $ crsctl setperm resource resource_name -u user:oracle:r-x
   ```

3. Start the resource:

   ```
   $ crsctl start resource resource_name
   ```

Application Placement Policies

A resource can be started on any server, subject to the placement policies, the resource start dependencies, and the availability of the action script on that server.

The `PLACEMENT` resource attribute determines how Oracle Clusterware selects a server on which to start a resource and where to relocate the resource after a server failure. The `HOSTING_MEMBERS` and `SERVER_POOLS` attributes determine eligible servers to host a resource and the `PLACEMENT` attribute further refines the placement of resources.

> **See Also:** Appendix B, "Oracle Clusterware Resource Reference" for more information about the `HOSTING_MEMBERS` and `SERVER_POOLS` resource attributes

The value of the `PLACEMENT` resource attribute determines how Oracle Clusterware places resources when they are added to the cluster or when a server fails. Together with either the `HOSTING_MEMBERS` or `SERVER_POOLS` attributes, you can configure how Oracle Clusterware places the resources in a cluster. When the value of the `PLACEMENT` attribute is:

- `balanced`: Oracle Clusterware uses any online server pool for placement. Less loaded servers are preferred to servers with greater loads. To measure how loaded a server is, Oracle Clusterware uses the `LOAD` resource attribute of the resources that are in an `ONLINE` state on the server. Oracle Clusterware uses the sum total of the `LOAD` values to measure the current server load.
- `favored`: If a value is assigned to either of the `HOSTING_MEMBERS`, `SERVER_POOLS`, or `SERVER_CATEGORY` resource attributes, then that value expresses a preference. If

HOSTING_MEMBERS is populated and either SERVER_POOLS or SERVER_CATEGORY is set, then HOSTING_MEMBERS indicates placement preference and SERVER_POOLS or SERVER_CATEGORY indicates a restriction. For example, the ora.cluster.vip resource has a policy that sets the value of PLACEMENT to favored, SERVER_CATEGORY is set to Hub, and HOSTING_MEMBERS is set to server_name1. In this case, Oracle Clusterware restricts the placement of ora.cluster.vip to the servers in the Hub category and then it prefers the server known as server_name1.

- restricted: Oracle Clusterware only considers servers that belong to server pools listed in the SEVER_POOLS resource attribute, servers of a particular category as configured in the SERVER_CATEGORY resource attribute, or servers listed in the HOSTING_MEMBERS resource attribute for resource placement. Only one of these resource attributes can have a value, otherwise it results in an error.

  See Also:

  - "HOSTING_MEMBERS" on page B-7
  - "SERVER_CATEGORY" on page B-9
  - "SERVER_POOLS" on page B-10

Unregistering Applications and Application Resources

To unregister a resource, use the crsctl delete resource command. You cannot unregister an application or resource that is ONLINE or required by another resource, unless you use the -force option. The following example unregisters the Apache Web server application:

```
$ crsctl delete resource myApache
```

Run the crsctl delete resource command as a clean-up step when a resource is no longer managed by Oracle Clusterware. Oracle recommends that you unregister any unnecessary resources.

Managing Resources

This section includes the following topics:

- Registering Application Resources
- Starting Application Resources
- Relocating Applications and Application Resources
- Stopping Applications and Application Resources
- Displaying Clusterware Application and Application Resource Status Information

Registering Application Resources

Each application that you manage with Oracle Clusterware is stored as a resource in OCR. Use the crsctl add resource command to register applications in OCR. For example, enter the following command to register the Apache Web server application from the previous example:

```
$ crsctl add resource myApache -type cluster_resource
-attr "ACTION_SCRIPT=/opt/cluster/scripts/myapache.scr, PLACEMENT=restricted,
SERVER_POOLS=server_pool_list,CHECK_INTERVAL=30,RESTART_ATTEMPTS=2,
START_DEPENDENCIES=hard(appsvip),STOP_DEPENDENCIES=hard(appsvip)"
```

If you modify a resource, then update OCR by running the `crsctl modify resource` command.

Starting Application Resources

Start and stop resources with the `crsctl start resource` and `crsctl stop resource` commands. Manually starting or stopping resources outside of Oracle Clusterware can invalidate the resource status. In addition, Oracle Clusterware may attempt to restart a resource on which you perform a manual stop operation.

To start an application resource that is registered with Oracle Clusterware, use the `crsctl start resource` command. For example:

```
$ crsctl start resource myApache
```

> **See Also:** Appendix E, "Oracle Clusterware Control (CRSCTL) Utility Reference" for usage information and examples of CRSCTL command output

The command waits to receive a notification of success or failure from the action program each time the action program is called. Oracle Clusterware can start application resources if they have stopped due to exceeding their failure threshold values. You must register a resource using `crsctl add resource` before you can start it.

Running the `crsctl start resource` command on a resource sets the resource `TARGET` value to `ONLINE`. Oracle Clusterware attempts to change the state to match the `TARGET` by running the action program with the `start` action.

If a cluster server fails while you are starting a resource on that server, then check the state of the resource on the cluster by using the `crsctl status resource` command.

Relocating Applications and Application Resources

Use the `crsctl relocate resource` command to relocate applications and application resources. For example, to relocate the Apache Web server application to a server named `rac2`, run the following command:

```
# crsctl relocate resource myApache -n rac2
```

Each time that the action program is called, the `crsctl relocate resource` command waits for the duration specified by the value of the `SCRIPT_TIMEOUT` resource attribute to receive notification of success or failure from the action program. A relocation attempt fails if:

- The application has required resources that run on the initial server
- Applications that require the specified resource run on the initial server

To relocate an application and its required resources, use the `-f` option with the `crsctl relocate resource` command. Oracle Clusterware relocates or starts all resources that are required by the application regardless of their state.

Stopping Applications and Application Resources

Stop application resources with the `crsctl stop resource` command. The command sets the resource `TARGET` value to `OFFLINE`. Because Oracle Clusterware always attempts to match the state of a resource to its target, the Oracle Clusterware subsystem stops the application. The following example stops the Apache Web server:

```
# crsctl stop resource myApache
```

You cannot stop a resource if another resource has a hard stop dependency on it, unless you use the force (`-f`) option. If you use the `crsctl stop resource` *`resource_name`* `-f` command on a resource upon which other resources depend, and if those resources are running, then Oracle Clusterware stops the resource *and* all of the resources that depend on the resource that you are stopping.

Displaying Clusterware Application and Application Resource Status Information

To display status information about applications and resources that are on cluster servers, use the `crsctl status resource` command. The following example displays the status information for the Apache Web server application:

```
# crsctl status resource myApache

NAME=myApache
TYPE=cluster_resource
TARGET=ONLINE
STATE=ONLINE on server010
```

Other information this command returns includes the following:

- How many times the resource has been restarted
- How many times the resource has failed within the failure interval
- The maximum number of times that a resource can restart or fail
- The target state of the resource and the normal status information

Use the `-f` option with the `crsctl status resource` *`resource_name`* command to view full information of a specific resource.

Enter the following command to view information about *all* applications and resources in tabular format:

```
# crsctl status resource -t
```

> **See Also:** Appendix E, "Oracle Clusterware Control (CRSCTL) Utility Reference" for detailed information about CRSCTL commands

Managing Automatic Restart of Oracle Clusterware Resources

You can prevent Oracle Clusterware from automatically restarting a resource by setting several resource attributes. You can also control how Oracle Clusterware manages the restart counters for your resources. In addition, you can customize the timeout values for the `start`, `stop`, and `check` actions that Oracle Clusterware performs on resources.

This section includes the following topics:

- Preventing Automatic Restarts
- Automatically Manage Restart Attempts Counter for Resources

Preventing Automatic Restarts

When a server restarts, Oracle Clusterware attempts to start the resources that run on the server as soon as the server starts. Resource startup might fail, however, if system components on which a resource depends, such as a volume manager or a file system, are not running. This is especially true if Oracle Clusterware does not manage the system components on which a resource depends. To manage automatic restarts, use the AUTO_START resource attribute to specify whether Oracle Clusterware should automatically start a resource when a server restarts.

Note: Regardless of the value of the AUTO_START resource attribute for a resource, the resource can start if another resource has a hard or weak start dependency on it or if the resource has a pullup start dependency on another resource.

See Also:

- "Start Dependencies" on page 9-14 for more information
- Appendix B, "Oracle Clusterware Resource Reference" for more information about the AUTO_START resource attribute

Automatically Manage Restart Attempts Counter for Resources

When a resource fails, Oracle Clusterware attempts to restart the resource the number of times specified in the RESTART_ATTEMPTS resource attribute, regardless of how often the resource fails. The CRSD process maintains an internal counter to track how often Oracle Clusterware restarts a resource. The number of times Oracle Clusterware has attempted to restart a resource is reflected in the RESTART_COUNT resource attribute. Oracle Clusterware can automatically manage the restart attempts counter based on the stability of a resource. The UPTIME_THRESHOLD resource attribute determines the time period that a resource must remain online, after which the RESTART_COUNT attribute gets reset to 0. In addition, the RESTART_COUNT resource attribute gets reset to 0 if the resource is relocated or restarted by the user, or the resource fails over to another server.

A

Cluster Verification Utility Reference

Cluster Verification Utility (CVU) performs system checks in preparation for installation, patch updates, or other system changes. Using CVU ensures that you have completed the required system configuration and preinstallation steps so that your Oracle Grid Infrastructure or Oracle Real Application Clusters (Oracle RAC) installation, update, or patch operation, completes successfully.

Oracle Universal Installer is fully integrated with CVU, automating many CVU prerequisite checks. Oracle Universal Installer runs all prerequisite checks and associated fixup scripts when you run the installer.

See Also:

- *Oracle Real Application Clusters Administration and Deployment Guide* for information about using the Server Control utility (SRVCTL) to manage CVU
- *Oracle Grid Infrastructure Installation Guide* and *Oracle Real Application Clusters Installation Guide* for information about how to manually install CVU

Note: Check for and download updated versions of CVU on Oracle Technology Network at

`http://www.oracle.com/technetwork/index.html`

This appendix describes CVU under the following topics:

- About Cluster Verification Utility
  - Overview
  - Operational Notes
  - Special Topics
- Cluster Verification Utility Command Reference
- Troubleshooting and Diagnostic Output for CVU

About Cluster Verification Utility

This section includes topics which relate to using CVU.

- Overview
- Operational Notes

- Special Topics

Overview

CVU can verify the primary cluster components during an operational phase or stage. A component can be basic, such as free disk space, or it can be complex, such as checking Oracle Clusterware integrity. For example, CVU can verify multiple Oracle Clusterware subcomponents across Oracle Clusterware layers. Additionally, CVU can check disk space, memory, processes, and other important cluster components. A stage could be, for example, database installation, for which CVU can verify whether your system meets the criteria for an Oracle Real Application Clusters (Oracle RAC) installation. Other stages include the initial hardware setup and the establishing of system requirements through the fully operational cluster setup.

Table A–1 lists verifications you can perform using CVU.

Table A–1 Performing Various CVU Verifications

Verification to Perform	CVU Commands to Use
System requirements verification	`cluvfy comp sys`
Oracle Cluster File System verification	`cluvfy stage [-pre | -post] cfs`
Storage verifications	■ `cluvfy comp space` ■ `cluvfy comp ssa` ■ `cluvfy stage [-pre | -post] acfscfg`
Network verification	`cluvfy stage -post hwos`
Connectivity verifications	■ `cluvfy comp nodecon` ■ `cluvfy comp nodereach`
Cluster Time Synchronization Services verification	`cluvfy comp clocksync`
User and Permissions verification	`cluvfy comp admprv`
Node comparison and verification	`cluvfy comp peer`
Installation verification	■ `cluvfy stage -pre dbcfg` ■ `cluvfy stage -pre dbinst` ■ `cluvfy stage [-pre | -post] crsinst` ■ `cluvfy stage [-pre | -post] hacfg` ■ `cluvfy stage [-pre | -post] nodeadd`
Deletion verification	`cluvfy stage -post nodedel`

Table A–1 (Cont.) Performing Various CVU Verifications

Verification to Perform	CVU Commands to Use
Oracle Clusterware and Oracle ASM Component verifications	■ `cluvfy comp acfs` ■ `cluvfy comp asm` ■ `cluvfy comp clumgr` ■ `cluvfy comp crs` ■ `cluvfy comp dhcp` ■ `cluvfy comp dns` ■ `cluvfy comp gns` ■ `cluvfy comp gpnp` ■ `cluvfy comp ha` ■ `cluvfy comp nodeapp` ■ `cluvfy comp ocr` ■ `cluvfy comp ohasd` ■ `cluvfy comp olr` ■ `cluvfy comp scan` ■ `cluvfy comp software` ■ `cluvfy comp vdisk`

Operational Notes

This section includes the following topics:

- Installation Requirements
- Usage Information
- CVU Configuration File
- Privileges and Security
- Using CVU Help

Installation Requirements

CVU installation requirements are:

- At least 30 MB free space for the CVU software on the node from which you run CVU
- A work directory with at least 25 MB free space on each node. The default location of the work directory is `/tmp` on Linux and UNIX systems, and the value specified in the `TEMP` environment variable on Windows systems. You can specify a different location by setting the `CV_DESTLOC` environment variable.

  When using CVU, the utility attempts to copy any needed information to the CVU work directory. It checks for the existence of the work directory on each node. If it does not find one, then it attempts to create one. Make sure that the CVU work directory either exists on all nodes in your cluster or proper permissions are established on each node for the user running CVU to create that directory.

Usage Information

CVU includes two scripts: `runcluvfy.sh` (`runcluvfy.bat` on Windows), which you use before installing Oracle software, and `cluvfy` (`cluvfy.bat` on Windows), located in the *`Grid_home`*`/bin` directory. The `runcluvfy.sh` script contains temporary variable definitions which enable it to run before you install Oracle Grid Infrastructure or Oracle Database. After you install Oracle Grid Infrastructure, use the `cluvfy` command to check prerequisites and perform other system readiness checks.

Note: Oracle Universal Installer runs `cluvfy` to check all prerequisites during Oracle software installation.

Before installing Oracle software, run `runcluvfy.sh` from the mountpoint path of the software installation media, as follows:

```
cd /mountpoint
./runcluvfy.sh options
```

In the preceding example, the *`options`* variable represents CVU command options that you select. For example:

```
$ cd /mnt/dvdrom
$ ./runcluvfy.sh comp nodereach -n node1,node2 -verbose
```

When you enter a CVU command, it provides a summary of the test. During preinstallation, Oracle recommends that you obtain detailed output by using the `-verbose` argument with the CVU command. The `-verbose` argument produces detailed output of individual checks. Where applicable, it shows results for each node in a tabular layout.

Run the CVU command-line tool using the `cluvfy` command. Using `cluvfy` does not adversely affect your cluster environment or your installed software. You can run `cluvfy` commands at any time, even before the Oracle Clusterware installation. In fact, CVU is designed to assist you as soon as your hardware and operating system are operational. If you run a command that requires Oracle Clusterware on a node, then CVU reports an error if Oracle Clusterware is not yet installed on that node.

The node list that you use with CVU commands should be a comma-delimited list of host names without a domain. CVU ignores domains while processing node lists. If a CVU command entry has duplicate node entries after removing domain information, then CVU eliminates the duplicate node entries.

For network connectivity verification, CVU discovers all of the available network interfaces if you do not specify an interface on the command line. For storage accessibility verification, CVU discovers shared storage for all of the supported storage types if you do not specify a particular storage identification on the command line. CVU also discovers the Oracle Clusterware home if one is available.

See Also: "Privileges and Security" on page A-5 for usage security information

CVU Configuration File

You can use the CVU configuration file to define specific inputs for the execution of CVU. The path for the configuration file is *`Grid_home`*`/cv/admin/cvu_config` (or *`Staging_area`*`\clusterware\stage\cvu\cv\admin` on Windows platforms). You can modify this file using a text editor. The inputs to CVU are defined in the form of key entries. You must follow these rules when modifying the CVU configuration file:

- Key entries have the syntax `name=value`
- Each key entry and the value assigned to the key only defines one property
- Lines beginning with the number sign (`#`) are comment lines and are ignored

- Lines that do not follow the syntax name=value are ignored

The following is the list of keys supported by CVU:

- CV_NODE_ALL: If set, it specifies the list of nodes that should be picked up when Oracle Clusterware is not installed. By default, this entry is commented out.
- CV_ORACLE_RELEASE: If set, it specifies the specific Oracle release (10.1, 10.2, 11.1, 11.2, or 12.1) for which the verifications have to be performed. If set, you do not have to use the -r *release* option wherever it is applicable. The default value is 12.1.
- CV_RAW_CHECK_ENABLED: If set to TRUE, it enables the check for accessibility of shared disks on Linux and UNIX systems. This shared disk accessibility check requires that you install the cvuqdisk RPM Package Manager (rpm) on all of the nodes. By default, this key is set to TRUE and shared disk check is enabled.
- CV_ASSUME_DISTID: This property is used in cases where CVU cannot detect or support a particular platform or a distribution. Oracle does not recommend that you change this property as this might render CVU non-functional.
- CV_XCHK_FOR_SSH_ENABLED: If set to TRUE, it enables the X-Windows check for verifying user equivalence with ssh. By default, this entry is commented out and X-Windows check is disabled.
- ORACLE_SRVM_REMOTECOPY: If set, it specifies the location for the scp or rcp command to override the CVU default value. By default, this entry is commented out and CVU uses /usr/bin/scp and /usr/sbin/rcp.
- ORACLE_SRVM_REMOTESHELL: If set, it specifies the location for ssh command to override the CVU default value. By default, this entry is commented out and the tool uses /usr/sbin/ssh.
- CV_ASSUME_CL_VERSION: By default, the command line parser uses crs activeversion for the display of command line syntax usage and syntax validation. Use this property to pass a version other than crs activeversion for command line syntax display and validation. By default, this entry is commented out.

If CVU does not find a key entry defined in the configuration file, then CVU searches for the environment variable that matches the name of the key. If the environment variable is set, then CVU uses its value, otherwise CVU uses a default value for that entity.

Privileges and Security

Because of a lack of user equivalence for the root user, most CVU commands cannot be run as root to perform any remote node operations, except for the following:

- cluvfy comp dhcp
- cluvfy comp dns

However, using privilege delegation, you can specify the -method parameter and choose one of two methods (sudo or root) to enable the checks and run the fixup scripts that require root privileges to be performed on remote nodes. You will be prompted for a password but the password is used dynamically while the CVU commands run, rather than being stored on a storage device.

Specifying the -method parameter is advantageous in the context of fixup scripts. If you choose privilege delegation, then all the fixup scripts can be run at one time from the local node. If you do not choose privilege delegation, then you must log onto each relevant node as root and run the fixup script.

Using CVU Help

The `cluvfy` commands have context sensitive help that shows their usage based on the command-line arguments that you enter. For example, if you enter `cluvfy`, then CVU displays high-level generic usage text describing the stage and component syntax. The following is a list of context help commands:

- `cluvfy -help`: CVU displays detailed CVU command information.
- `cluvfy -version`: CVU displays the version of Oracle Clusterware.
- `cluvfy comp -list`: CVU displays a list of components that can be checked, and brief descriptions of how the utility checks each component.
- `cluvfy comp -help`: CVU displays detailed syntax for each of the valid component checks.
- `cluvfy stage -list`: CVU displays a list of valid stages.
- `cluvfy stage -help`: CVU displays detailed syntax for each of the valid stage checks.

You can also use the `-help` option with any CVU command. For example, `cluvfy stage -pre nodeadd -help` returns detailed information for that particular command.

If you enter an invalid CVU command, then CVU shows the correct usage for that command. For example, if you type `cluvfy stage -pre dbinst`, then CVU shows the correct syntax for the precheck commands for the `dbinst` stage. Enter the `cluvfy -help` command to see detailed CVU command information.

Special Topics

This section includes the following topics:

- Generating Fixup Scripts
- Using CVU to Determine if Installation Prerequisites are Complete
- Using CVU with Oracle Database 10g Release 1 or 2
- Entry and Exit Criteria
- Verbose Mode and UNKNOWN Output
- CVU Node List Shortcuts

Generating Fixup Scripts

You can use the `-fixup` flag with certain CVU commands to generate **fixup scripts** before installation. Oracle Universal Installer can also generate fixup scripts during installation. The installer then prompts you to run the script as `root` in a separate terminal session. If you generate a fixup script from the command line, then you can run it as `root` after it is generated. When you run the script, it raises kernel values to required minimums, if necessary, and completes other operating system configuration.

Alternatively, you can specify the `-method` parameter with certain CVU commands to enable privilege delegation and enable you to run fixup scripts as `root` on remote nodes from the local node.

See Also: "Privileges and Security" on page A-5

Using CVU to Determine if Installation Prerequisites are Complete

You can use CVU to determine which system prerequisites for installation are completed. Use this option if you are installing Oracle Database 12*c* software on a system with a pre-existing Oracle software installation. In using this option, note the following:

- You must run CVU as the user account you plan to use to run the installation. You cannot run CVU as `root`, and running CVU as another user other than the user that is performing the installation does not ensure the accuracy of user and group configuration for installation or other configuration checks.
- Before you can complete a clusterwide status check, SSH must be configured for all cluster nodes. You can use the installer to complete SSH configuration, or you can complete SSH configuration yourself between all nodes in the cluster. You can also use CVU to generate a fixup script to configure SSH connectivity.
- CVU can assist you by finding preinstallation steps that must be completed, but it cannot perform preinstallation tasks.

Use the following syntax to determine what preinstallation steps are completed, and what preinstallation steps you must perform; running the command with the `-fixup` flag generates a fixup script to complete kernel configuration tasks as needed:

```
$ ./runcluvfy.sh stage -pre crsinst -fixup -n node_list
```

In the preceding syntax example, replace the *`node_list`* variable with the names of the nodes in your cluster, separated by commas. On Windows, you must enclose the comma-delimited node list in double quotation marks (`""`).

For example, for a cluster with mountpoint `/mnt/dvdrom/`, and with nodes `node1`, `node2`, and `node3`, enter the following command:

```
$ cd /mnt/dvdrom/
$ ./runcluvfy.sh stage -pre crsinst -fixup -n node1,node2,node3
```

Review the CVU report, and complete additional steps as needed.

> **See Also:** Your platform-specific installation guide for more information about installing your product

Using CVU with Oracle Database 10*g* Release 1 or 2

You can use CVU included on the Oracle Database 12*c* media to check system requirements for Oracle Database 10*g* release 1 (10.1) and later installations. To use CVU to check Oracle Clusterware installations, append the command `-r` *`release_code`* flag to the standard CVU system check commands.

For example, to perform a verification check before installing Oracle Clusterware version 10. 2 on a system where the media mountpoint is `/mnt/dvdrom` and the cluster nodes are `node1`, `node2`, and `node3`, enter the following command:

```
$ cd /mnt/dvdrom
$ ./runcluvfy.sh stage -pre crsinst -n node1,node2,node3 -r 10.2
```

> **Note:** If you do not specify a release version to check, then CVU checks for 12*c* release 1 (12.1) requirements.

Entry and Exit Criteria

When verifying stages, CVU uses entry and exit criteria. Each stage has entry criteria that define a specific set of verification tasks to be performed before initiating that stage. This check prevents you from beginning a stage, such as installing Oracle Clusterware, unless you meet the Oracle Clusterware prerequisites for that stage.

The exit criteria for a stage define another set of verification tasks that you must perform after the completion of the stage. Post-checks ensure that the activities for that stage have been completed. Post-checks identify stage-specific problems before they propagate to subsequent stages.

Verbose Mode and UNKNOWN Output

Although by default CVU reports in nonverbose mode by only reporting the summary of a test, you can obtain detailed output by using the `-verbose` argument. The `-verbose` argument produces detailed output of individual checks and where applicable shows results for each node in a tabular layout.

If a `cluvfy` command responds with `UNKNOWN` for a particular node, then this is because CVU cannot determine whether a check passed or failed. The cause could be a loss of reachability or the failure of user equivalence to that node. The cause could also be any system problem that was occurring on that node when CVU was performing a check.

The following is a list of possible causes for an `UNKNOWN` response:

- The node is down
- Executables that CVU requires are missing in *`Grid_home`*`/bin` or the *`Oracle home`* directory
- The user account that ran CVU does not have privileges to run common operating system executables on the node
- The node is missing an operating system patch or a required package
- The node has exceeded the maximum number of processes or maximum number of open files, or there is a problem with IPC segments, such as shared memory or semaphores

CVU Node List Shortcuts

To provide CVU a list of all of the nodes of a cluster, enter `-n all`. CVU attempts to obtain the node list in the following order:

1. If vendor clusterware is available, then CVU selects all of the configured nodes from the vendor clusterware using the `lsnodes` utility.
2. If Oracle Clusterware is installed, then CVU selects all of the configured nodes from Oracle Clusterware using the `olsnodes` utility.
3. If neither the vendor clusterware nor Oracle Clusterware is installed, then CVU searches for a value for the `CV_NODE_ALL` key in the configuration file.
4. If vendor clusterware and Oracle Clusterware are not installed and no key named `CV_NODE_ALL` exists in the configuration file, then CVU searches for a value for the `CV_NODE_ALL` environmental variable. If you have not set this variable, then CVU reports an error.

To provide a partial node list, you can set an environmental variable and use it in the CVU command. For example, on Linux or UNIX systems you can enter:

```
setenv MYNODES node1,node3,node5
cluvfy comp nodecon -n $MYNODES [-verbose]
```

Cluster Verification Utility Command Reference

This section lists and describes the following CVU commands:

- `cluvfy comp acfs`

- `cluvfy comp admprv`
- `cluvfy comp asm`
- `cluvfy comp baseline`
- `cluvfy comp clocksync`
- `cluvfy comp clumgr`
- `cluvfy comp crs`
- `cluvfy comp dhcp`
- `cluvfy comp dns`
- `cluvfy comp freespace`
- `cluvfy comp gns`
- `cluvfy comp gpnp`
- `cluvfy comp ha`
- `cluvfy comp healthcheck`
- `cluvfy comp nodeapp`
- `cluvfy comp nodecon`
- `cluvfy comp nodereach`
- `cluvfy comp ocr`
- `cluvfy comp ohasd`
- `cluvfy comp olr`
- `cluvfy comp peer`
- `cluvfy comp scan`
- `cluvfy comp software`
- `cluvfy comp space`
- `cluvfy comp ssa`
- `cluvfy comp sys`
- `cluvfy comp vdisk`
- `cluvfy stage [-pre | -post] acfscfg`
- `cluvfy stage [-pre | -post] cfs`
- `cluvfy stage [-pre | -post] crsinst`
- `cluvfy stage -pre dbcfg`
- `cluvfy stage -pre dbinst`
- `cluvfy stage [-pre | -post] hacfg`
- `cluvfy stage -post hwos`
- `cluvfy stage [-pre | -post] nodeadd`
- `cluvfy stage -post nodedel`

cluvfy comp acfs

Use the `cluvfy comp acfs` component verification command to check the integrity of Oracle Automatic Storage Management Cluster File System (Oracle ACFS) on all nodes in a cluster.

Syntax

```
cluvfy comp acfs [-n node_list] [-f file_system] [-verbose]
```

Parameters

Table A–2 cluvfy comp acfs Command Parameters

Parameter	Description
`-n node_list`	The comma-delimited list of nondomain qualified node names on which to conduct the verification. If you do not specify this option, then CVU checks only the local node.
`-f file_system`	The name of the file system to check.
`-verbose`	CVU prints detailed output.

cluvfy comp admprv

Use the `cluvfy comp admprv` command to check the required administrative privileges for the operation specified by `-o` parameter on all the nodes that you specify in the node list. The operations are mutually exclusive and you can specify only one operation at a time.

Syntax

On Linux and UNIX platforms:

```
cluvfy comp admprv [-n node_list] -o user_equiv [-sshonly] |
  -o crs_inst [-asmgrp asmadmin_group] [-asmdbagrp asmdba_group] [-orainv
orainventory_group]
  [-fixup] [-fixupnoexec] [-method {sudo -user user_name [-location directory_
path] | root}]
  | -o db_inst [-osdba osdba_group] [-fixup] [-fixupnoexec]
  [-method {sudo -user user_name [-location dir_path] | root}] |
  -o db_config -d oracle_home [-fixup] [-fixupnoexec]
  [-method {sudo -user user_name [-location directory_path] | root}] [-verbose]
```

On Windows platforms:

```
cluvfy comp admprv [-n node_list] | -o user_equiv | -o crs_inst [-fixup]
[-fixupnoexec]
  | -o db_inst [-fixup] [-fixupnoexec] | -o db_config -d oracle_home [-fixup]
[-fixupnoexec]
 [-verbose]
```

Parameters

Table A–3 cluvfy comp admprv Command Parameters

Parameter	Description
`-n` *node_list*	The comma-delimited list of nondomain qualified node names on which to conduct the verification. If you do not specify this option, then CVU checks only the local node.
`-o user_equiv [-sshonly]`	Checks user equivalence between the nodes. On Linux and UNIX platforms, this command verifies user equivalence using `ssh`.

`-o crs_inst [-asmgrp` *asmadmin_group*`]`
`[-asmdbagrp` *asmdba_group*`]`
`[-orainv` *orainventory_group*`]`
	Checks administrative privileges for installing Oracle Clusterware. Optionally, you can specify the following:
	`-asmgrp`: Specify the name of the OSASM group. The default is `asmadmin`.
	`-asmdbagrp`: Specify the name of the ASMDBA group. The default is `asmdba`.
	`-orainv`: Specify the name of the Oracle Inventory group. The default is `oinstall`.

`-o db_inst [-osdba` *osdba_group*`]`	Checks administrative privileges for installing an Oracle RAC database. Optionally, you can specify the following: `-osdba`: The name of the OSDBA group. The default is `dba`.
`-o db_config -d` *oracle_home*	Checks administrative privileges for creating or configuring an Oracle RAC database. Specify the location of the Oracle home for the Oracle RAC database.
`-fixup`	Specifies that if the verification fails, then CVU generates fixup instructions, if feasible.
`-fixupnoexec`	Specifies that if verification fails, then CVU generates the fixup data and displays the instructions for manual execution of the generated fixups.
`-method {sudo -user` *user_name* `[-location` *dir_path*`] | root}`	Specify whether the privilege delegation method is `sudo` or `root`, for root user access. If you specify `sudo`, then you must specify the user name to access all the nodes with root privileges and, optionally, provide the full file system path for the `sudo` executable.
`-verbose`	CVU prints detailed output.

Usage Notes

- By default, the equivalence check does not verify X-Windows configurations, such as whether you have disabled X-forwarding, whether you have the proper setting for the `DISPLAY` environment variable, and so on.

  To verify X-Windows aspects during user equivalence checks, set the `CV_XCHK_FOR_SSH_ENABLED` key to `TRUE` in the configuration file that resides in the `CV_HOME/cv/admin/cvu_config` directory before you run the `cluvfy comp admprv -o user_equiv` command.

Examples

Example 1: Verifying Permissions Required to Install Oracle Clusterware

You can verify that the permissions required for installing Oracle Clusterware have been configured on the nodes `racnode1` and `racnode2` by running the following command:

```
cluvfy comp admprv -n racnode1,racnode2 -o crs_inst -verbose
```

Example 1: Verifying Permissions Manage Oracle RAC Databases

You can verify that the permissions required for creating or modifying an Oracle RAC database using the `C:\app\oracle\product\11.2.0\dbhome_1` Oracle home directory, and generate a script to configure the permissions by running the following command:

```
cluvfy comp admprv -n racnode1,racnode2 -o db_config -d
C:\app\oracle\product\11.2.0\dbhome_1 -fixup -verbose
```

cluvfy comp asm

Use the `cluvfy comp asm` component verification command to check the integrity of Oracle Automatic Storage Management (Oracle ASM) on specific nodes in the cluster. This check ensures that the Oracle ASM instances on the specified nodes are running from the same Oracle home and that `asmlib`, if it exists, has a valid version and ownership.

Syntax

```
cluvfy comp asm [-n node_list] [-verbose]
```

Parameters

Table A–4 cluvfy comp asm Command Parameters

Parameter	Description
`-n` *`node_list`*	The comma-delimited list of nondomain qualified node names on which to conduct the verification. If you do not specify this option, then CVU checks only the local node.
`-verbose`	CVU prints detailed output.

Examples

Verifying the Integrity of Oracle ASM on All Nodes

To verify the integrity of Oracle ASM on all of the nodes in the cluster, use the following command:

```
cluvfy comp asm -n all
```

This command produces output similar to the following:

```
Verifying ASM Integrity

Task ASM Integrity check started...

Starting check to see if ASM is running on all cluster nodes...

ASM Running check passed. ASM is running on all specified nodes

Starting Disk Groups check to see if at least one Disk Group configured...
Disk Group Check passed. At least one Disk Group configured

Task ASM Integrity check passed...

Verification of ASM Integrity was successful.
```

cluvfy comp baseline

Use the `cluvfy comp baseline` command to capture system and cluster configuration information to create a baseline. You can use this baseline for comparison with the state of the system. You can collect baselines at strategic times, such as after Oracle Clusterware installation, before and after upgrading Oracle Clusterware, or automatically as part of periodic execution of CVU running as an Oracle Clusterware resource. You can also compare several baselines.

Syntax

```
cluvfy comp baseline -collect {all | cluster | database} [-n node_list]
  [-d Oracle_home] [-db db_unique_name] [-bestpractice | -mandatory]
[-binlibfilesonly
  [-reportname report_name] [-savedir save_dir]

cluvfy comp baseline -compare baseline1,baseline2,... [-deviations]
```

Parameters

Table A–5 cluvfy comp baseline Command Parameters

Parameter	Description
`-collect {all | cluster | database}`	The `-collect` parameter instructs CVU to create a baseline and save it in the *Grid_home*`/cv/report/xml` directory. You can collect a baseline related to Oracle Clusterware (`cluster`), the database (`database`), or both (`all`).
`-n` *node_list*	Specify a comma-delimited list of non-domain qualified node names on which the test should be conducted.
`-d` *Oracle_home*	When collecting a database baseline, if you specify an Oracle home, then CVU collects baselines for all the databases running from the Oracle home. Use the `-db` parameter to collect a baseline for a specific database.
`-db` *db_unique_name*	The name of the database for which you want to collect information. When collecting a database baseline, if you specify the `-db` parameter, then CVU only collects the baseline for the specified database. If you do not specify `-db`, then CVU discovers all of the cluster databases configured in the system and the collects baselines for each of those.
`-bestpractice | -mandatory`	Specify `-bestpractice` to collect a baseline for only best practice recommendations. Specify `-mandatory` to collect a baseline for only mandatory requirements.
`-binlibfilesonly`	Specify `-binlibfilesonly` to collect only files in the `bin/`, `lib/`, and `jlib/` subdirectories of the software home.
`-report` *report_name*	Use this optional parameter to specify a name for the report.
`-savedir` *save_dir*	Use this optional parameter to specify a location in which CVU saves the reports. If you do not specify the `-savedir` option, then CVU saves the reports in the *Grid_home*`/cv/report` directory.

Table A–5 (Cont.) cluvfy comp baseline Command Parameters

Parameter	Description
`-compare` *`baseline1`*`,`*`baseline2`*`,...`	Specify `-compare` to compare baselines. If you specify only one baseline, then CVU displays the results of the collections. If you specify multiple baselines in a comma-delimited list, then CVU compares the values from the baselines against each other in an HTML document.
`-deviations`	Specify this parameter to display only the deviations from best practice recommendations or mandatory requirements, or both, (depending on whether you specified the `-bestpractice` and `-mandatory` parameters).

Usage Notes

- Specify either the `-collect` or `-compare` parameter.
- Items that CVU collects when running this command include:

Physical memory
Available memory
Swap space
Free space
Required packages
Recommended kernel parameters
`/etc/inittab` permissions
Domain sockets under `/var/tmp/.oracle`
Oracle Clusterware software file attributes
Network MTU size
OCR permissions, group, and owner (if OCR is stored on a shared file system)
OCR disk group (if OCR is stored on Oracle ASM
System requirement pluggable tasks (Zeroconf settings, `/boot` mount, Huge Pages existence, 8888 port availability, Ethernet jumbo frames)
Oracle Clusterware post-check pluggable tasks (`css miscount`, `reboottime`, `disktimeout`)
Database best practices

Examples

The following examples illustrate usage for both `-collect` and `-compare` command parameters:

```
$ cluvfy comp baseline -collect all -n all -db orcl -bestpractice -report bl1
   -savedir /tmp

$ cluvfy comp baseline -compare bl1,bl2
```

cluvfy comp clocksync

Use the `cluvfy comp clocksync` component verification command to clock synchronization across all the nodes in the node list. CVU verifies a time synchronization service is running (Oracle Cluster Time Synchronization Service (CTSS) or Network Time Protocol (NTP)), that each node is using the same reference server for clock synchronization, and that the time offset for each node is within permissible limits.

Syntax

```
cluvfy comp clocksync [-noctss] [-n node_list] [-verbose]
```

Parameters

Table A–6 cluvfy comp clocksync Command Parameters

Parameter	Description
`-noctss`	If you specify this option, then CVU does not perform a check on CTSS. Instead, CVU checks the platform's native time synchronization service, such as NTP.
`-n node_list`	The comma-delimited list of nondomain qualified node names on which to conduct the verification. If you do not specify this option, then CVU checks only the local node.
`-verbose`	CVU prints detailed output.

cluvfy comp clumgr

Use the `cluvfy comp clumgr` component verification command to check the integrity of cluster manager subcomponent, or Oracle Cluster Synchronization Services (CSS), on the nodes in the node list.

Syntax

```
cluvfy comp clumgr [-n node_list] [-verbose]
```

Parameters

Table A–7 cluvfy comp clumgr Command Parameters

Parameter	Description
`-n node_list`	The comma-delimited list of nondomain qualified node names on which to conduct the verification. If you do not specify this option, then CVU checks only the local node.
`-verbose`	CVU prints detailed output.

cluvfy comp crs

Run the `cluvfy comp crs` component verification command to check the integrity of the Cluster Ready Services (CRS) daemon on the specified nodes.

Syntax

```
cluvfy comp crs [-n node_list] [-verbose]
```

Parameters

Table A–8 cluvfy comp crs Command Parameters

Parameter	Description
`-n` *node_list*	The comma-delimited list of nondomain qualified node names on which to conduct the verification. If you do not specify this option, then CVU checks only the local node.
`-verbose`	CVU prints detailed output.

cluvfy comp dhcp

Use the `cluvfy comp dhcp` component verification command to verify that the DHCP server exists on the network, and that it can provide a required number of IP addresses. The required number of IP addresses is calculated, as follows:

- Regardless of the size of the cluster, there must be three scan VIPs
- One node VIP for each node you specify with the `-n` option
- One application VIP for each application VIP resource you specify with the `-vipresname` option

This verification also verifies the response time for the DHCP server.

Syntax

```
cluvfy comp dhcp -clustername cluster_name [-vipresname application_vip_resource_
name]
  [-port dhcp_port] [-n node_list] [-method {sudo -user user_name [-location
directory_path] | root}]
  [-networks network_list] [-verbose]
```

Parameters

Table A–9 cluvfy comp dhcp Command Parameters

Parameter	Description
`-clustername` *cluster_name*	The name of the cluster of which you want to check the integrity of DHCP.
`-vipresname` *application_vip_resource_name*	A comma-delimited list of the names of the application VIP resource.
`-port` *dhcp_port*	The port to which DHCP packages are sent. The default port is 67.
`-n` *node_list*	The comma-delimited list of nondomain qualified node names on which to conduct the verification. If you do not specify this option, then CVU checks only the local node.
`-method {sudo -user` *user_name* `[-location` *directory_path*`] | root}`	Specify whether the privilege delegation method is `sudo` or `root`, for root user access. If you specify `sudo`, then you must specify the user name to access all the nodes with root privileges and, optionally, provide the full file system path for the `sudo` executable.

Table A–9 (Cont.) cluvfy comp dhcp Command Parameters

Parameter	Description
`-networks` *network_list*	Specify a list of network classifications for the cluster separated by forward slashes (`/`) that you want CVU to check, where each network is in the form of `"`*if_ name*`"[:`*subnet_id*`[:`*if_type*`[,`*if_type*`...]]]`. In the preceding format, you must enclose *if_name* in double quotation marks, and you can use regular expressions, such as `".*"`, as in `"eth*"`, to match interfaces like `eth1` and `eth02`. The *subnet_id* is the subnet number of the network interface. The *if_type* is a comma-separated list of interface types: `{CLUSTER_INTERCONNECT | PUBLIC | ASM}`.
`-verbose`	CVU prints detailed output.

Usage Notes

- You must run this command as `root`.
- Do not run this check while the default network Oracle Clusterware resource, configured to use a DHCP-provided IP address, is online (because the VIPs get released and, since the cluster is online, DHCP has provided IP, so there is no need to double the load on the DHCP server).
- Run this check on the local node. This is unlike other CVU commands, which run on all nodes specified in node list. As a result, even though the local node is not included in the node list you specify with the `-n` option, the error messages get reported to the local node.
- Before running this command, ensure that the network resource is offline. Use the `srvctl stop nodeapps` command to bring the network resource offline, if necessary.

  See Also: *Oracle Real Application Clusters Administration and Deployment Guide* for more information about the `srvctl stop nodeapps` command

cluvfy comp dns

Use the `cluvfy comp dns` component verification command to verify that the Grid Naming Service (GNS) subdomain delegation has been properly set up in the Domain Name Service (DNS) server.

Note: Oracle does not support this command on Windows.

Syntax

```
cluvfy comp dns -server -domain gns_sub_domain -vipaddress gns_vip_address [-port
dns_port]
  [-method {sudo -user user_name [-location directory_path] | root}] [-verbose]

cluvfy comp dns -client -domain gns_sub_domain -vip gns_vip [-port dns_port]
  [-last] [-method {sudo -user user_name [-location directory_path] | root}]
[-verbose]
```

Parameters

Table A–10 cluvfy comp dns Command Parameters

Parameter	Description
`-server`	Start a test DNS server for the GNS subdomain that listens on the domain specified by the `-domain` option.
`-client`	Validate connectivity to a test DNS server started on a specific address. You must specify the same information you specified when you started the DNS server.
`-domain` *`gns_sub_domain`*	The GNS subdomain name.
`-vipaddress` *`gns_vip_address`*	GNS virtual IP address in the form `{`*`ip_name`*` | `*`ip_address`*`}/`*`net_mask`*`/`*`interface_name`*. You can specify either *`ip_name`*, which is a name that resolves to an IP address, or *`IP_address`*, which is an IP address. Either name or address is followed by *`net_mask`*, which is the subnet mask for the IP address, and *`interface_name`*, which is the interface on which to start the IP address.
`-vip` *`gns_vip`*	GNS virtual IP address, which is either a name that resolves to an IP address or a dotted decimal numeric IP address.
`-port` *`dns_port`*	The port on which the test DNS server listens. The default port is 53.
`-last`	Send a termination request to the test DNS server after all the validations are complete.
`-method {sudo -user` *`user_name`* `[-location` *`directory_path`*`] | root}`	Specify whether the privilege delegation method is `sudo` or `root`, for root user access. If you specify `sudo`, then you must specify the user name to access all the nodes with root privileges and, optionally, provide the full file system path for the `sudo` executable.
`-verbose`	CVU prints detailed output.

Usage Notes

- You must run this command as `root`.
- Run `cluvfy comp dns -server` on one node of the cluster.
- Run `cluvfy comp dns -client` on each node of the cluster to verify DNS server setup for the cluster.
- On the last node, specify the `-last` option to terminate the `cluvfy comp dns -server` instance.
- Do not run this command while the GNS resource is online.

cluvfy comp freespace

Use the `cluvfy comp freespace` component verification command to check the free space available in the Oracle Clusterware home storage and ensure that there is at least 5% of the total space available. For example, if the total storage is 10GB, then the check ensures that at least 500MB of it is free.

Syntax

```
cluvfy comp freespace [-n node_list]
```

If you choose to include the `-n` option, then enter a comma-delimited list of node names on which to run the command.

cluvfy comp gns

Use the `cluvfy comp gns` component verification command to verify the integrity of the Grid Naming Service (GNS) on the cluster.

Syntax

```
cluvfy comp gns -precrsinst {-vip gns_vip [-domain gns_domain] | -clientdata file_
name}
  [-networks network_list] [-n node_list] [-verbose]

cluvfy comp gns -postcrsinst [-verbose]
```

Parameters

Table A–11 cluvfy comp gns Command Parameters

Parameter	Description
`-precrsinst`	Perform checks on GNS domain name and VIP address before Oracle Clusterware is installed.
`-vip` *`gns_vip`*	Specify the GNS virtual IP address. When you specify `-vip` and `-domain` together, CVU validates that this cluster can become a GNS server (local GNS).
`-domain` *`gns_domain`*	Optionally, you can specify the GNS sub domain name.
`-clientdata` *`file_name`*	Specify the name of the file that contains the GNS credentials. CVU validates that this cluster can use the specified client data to become a client GNS cluster of another GNS server cluster (shared GNS).
`-networks` *`network_list`*	Specify a list of network classifications for the cluster, including public networks for GNS, separated by forward slashes (`/`) that you want CVU to check, where each network is in the form of `"`*`if_name`*`"[:`*`subnet_id`*`[:`*`if_type`*`[,`*`if_type`*`...]]]`. In the preceding format, you must enclose *`if_name`* in double quotation marks, and you can use regular expressions, such as `".*"`, as in `"eth*"`, to match interfaces like `eth1` and `eth02`. The *`subnet_id`* is the subnet number of the network interface. The *`if_type`* is a comma-separated list of interface types: `{CLUSTER_INTERCONNECT | PUBLIC | ASM}`.
`-n` *`node_list`*	The comma-delimited list of nondomain qualified node names on which to conduct the verification. If you do not specify this option, then CVU checks only the local node.
`-postcrsinst`	Check the integrity of GNS on all nodes in the cluster
`-verbose`	CVU prints detailed output.

cluvfy comp gpnp

Use the `cluvfy comp gpnp` component verification command to check the integrity of Grid Plug and Play on a list of nodes in a cluster.

Syntax

```
cluvfy comp gpnp [-n node_list] [-verbose]
```

Parameters

Table A–12 cluvfy comp gpnp Command Parameters

Parameter	Description
`-n` *node_list*	The comma-delimited list of nondomain qualified node names on which to conduct the verification. If you do not specify this option, then CVU checks only the local node.
`-verbose`	CVU prints detailed output.

cluvfy comp ha

Use the `cluvfy comp ha` component verification command to check the integrity of Oracle Restart on the local node.

Syntax

```
cluvfy comp ha [-verbose]
```

If you include the `-verbose` option, then CVU prints detailed output.

cluvfy comp healthcheck

Use the `cluvfy comp healthcheck` component verification command to check your Oracle Clusterware and Oracle Database installations for their compliance with mandatory requirements and best practices guidelines, and to ensure that they are functioning properly.

Syntax

```
cluvfy comp healthcheck [-collect {cluster|database}] [-db db_unique_name]
  [-bestpractice|-mandatory] [-deviations] [-html] [-save [-savedir directory_
path]]
```

Parameters

Table A–13 cluvfy comp healthcheck Command Parameters

Parameter	Description
`-collect {cluster|database}`	Use `-collect` to specify that you want to perform checks for Oracle Clusterware (`cluster`) or Oracle Database (`database`). If you do not use the `-collect` flag with the `healthcheck` option, then CVU performs checks for both Oracle Clusterware and Oracle Database.

Table A–13 (Cont.) cluvfy comp healthcheck Command Parameters

Parameter	Description
`-db` *db_unique_name*	Use `-db` to specify checks on the specific database that you enter after the `-db` flag. CVU uses JDBC to connect to the database as the user `cvusys` to verify various database parameters. For this reason, if you want CVU to perform checks for the database you specify with the `-db` flag, then you must first create the `cvusys` user on that database, and grant that user the CVU-specific role, `cvusapp`. You must also grant members of the `cvusapp` role `select` permissions on system tables. There is a SQL script included in the *CVU_home*`/cv/admin/cvusys.sql` directory to facilitate the creation of this user. Use this SQL script to create the `cvusys` user on all the databases that you want to verify using CVU. If you use the `-db` flag but do not provide a database unique name, then CVU discovers all the Oracle Databases on the cluster. To perform best practices checks on these databases, you must create the `cvusys` user on each database, and grant that user the `cvusapp` role with the `select` privileges needed to perform the best practice checks.
`[-bestpractice|-mandatory] [-deviations]`	Use the `-bestpractice` flag to specify best practice checks, and the `-mandatory` flag to specify mandatory checks. Add the `-deviations` flag to specify that you want to see only the deviations from either the best practice recommendations or the mandatory requirements. You can specify either the `-bestpractice` or `-mandatory` flag, but not both flags. If you specify neither `-bestpractice` nor `-mandatory`, then CVU displays both best practices and mandatory requirements.
`-html`	Use the `-html` flag to generate a detailed report in HTML format. If you specify the `-html` flag, and a browser CVU recognizes is available on the system, then CVU starts the browser and displays the report on the browser when the checks are complete. If you do not specify the `-html` flag, then CVU generates the detailed report in a text file.
`-save [-savedir` *directory_path*`]`	Use the `-save` or `-save -savedir` flags to save validation reports (`cvuchecdkreport_`*timestamp*`.txt` and `cvucheckreport_`*timestamp*`.htm`), where *timestamp* is the time and date of the validation report. If you use the `-save` flag by itself, then CVU saves the reports in the *CVU_home*`/cv/report` directory, where *CVU_home* is the location of the CVU binaries. If you use the `-save -savedir` flags, then specify a directory where you want CVU to save the reports.

cluvfy comp nodeapp

Use the component `cluvfy comp nodeapp` command to check for the existence of node applications, namely VIP, NETWORK, and ONS, on all of the specified nodes.

Syntax

```
cluvfy comp nodeapp [-n node_list] [-verbose]
```

Parameters

Table A–14 cluvfy comp nodeapp Command Parameters

Parameter	Description
`-n node_list`	The comma-delimited list of nondomain qualified node names on which to conduct the verification. If you do not specify this option, then CVU checks only the local node.
`-verbose`	CVU prints detailed output.

cluvfy comp nodecon

Use the `cluvfy comp nodecon` component verification command to check the connectivity among the nodes specified in the node list. If you provide an interface list, then CVU checks the connectivity using only the specified interfaces.

Syntax

```
cluvfy comp nodecon [-n node_list] [-networks network_list] [-verbose]
```

Parameters

Table A–15 cluvfy comp nodecon Command Parameters

Parameter	Description
`-n node_list`	The comma-delimited list of nondomain qualified node names on which to conduct the verification.
`-networks network_list`	The forward slash (/)-delimited list of networks on which to conduct the verification.
`-verbose`	CVU prints detailed output.

Usage Notes

- You can run this command in verbose mode to identify the mappings between the interfaces, IP addresses, and subnets.
- On Solaris platforms, this command skips testing IP addresses that are marked as deprecated.
- Use the `nodecon` command without the `-networks` option and with `-n` set to `all` to use CVU to:
  - Discover all of the network interfaces that are available on the cluster nodes
  - Review the interfaces' corresponding IP addresses and subnets
  - Obtain the list of interfaces that are suitable for use as VIPs and the list of interfaces to private interconnects
  - Verify the connectivity between all of the nodes through those interfaces

Examples

Example 1: Verifying the connectivity between nodes through specific network interfaces:

You can verify the connectivity between the nodes `node1` and `node3` through interface `eth0` by running the following command:

```
cluvfy comp nodecon -n node1/node3 -networks eth0 -verbose
```

Example 2: Discovering all available network interfaces and verifying the connectivity between the nodes in the cluster through those network interfaces:

Use the following command to discover all of the network interfaces that are available on the cluster nodes. CVU then reviews the interfaces' corresponding IP addresses and subnets. Using this information, CVU obtains a list of interfaces that are suitable for use as VIPs and a list of interfaces to private interconnects. Finally, CVU verifies the connectivity between all of the nodes in the cluster through those interfaces.

```
cluvfy comp nodecon -n all -verbose
```

cluvfy comp nodereach

Use the `cluvfy comp nodereach` component verification command to check the reachability of specified nodes from a source node.

Syntax

```
cluvfy comp nodereach -n node_list [-srcnode node] [-verbose]
```

Parameters

Table A–16 cluvfy comp nodereach Command Parameters

Parameter	Description
`-n` *node_list*	The comma-delimited list of nondomain qualified node names on which to conduct the verification.
`-srcnode` *node*	The name of the source node from which CVU performs the reachability test. If you do not specify a source node, then the node on which you run the command is used as the source node.
`-verbose`	CVU prints detailed output.

Example

Verifying the network connectivity between nodes in the cluster:

To verify that `node3` is reachable over the network from the local node, use the following command:

```
cluvfy comp nodereach -n node3
```

This command produces output similar to the following:

```
Verifying node reachability

Checking node reachability...
Node reachability check passed from node “node1”


Verification of node reachability was successful.
```

cluvfy comp ocr

Use the `cluvfy comp ocr` component verification command to check the integrity of Oracle Cluster Registry (OCR) on all the specified nodes.

Syntax

```
cluvfy comp ocr [-n node_list] [-method {sudo -user user_name [-location
directory_path] | root}]
   [-verbose]
```

Parameters

Table A–17 cluvfy comp ocr Command Parameters

Parameter	Description
`-n node_list`	The comma-delimited list of nondomain qualified node names on which to conduct the verification. If you do not specify this option, then CVU checks only the local node.
`-method {sudo -user user_name [-location directory_path] | root}`	Specify whether the privilege delegation method is `sudo` or `root`, for root user access. If you specify `sudo`, then you must specify the user name to access all the nodes with root privileges and, optionally, provide the full file system path for the `sudo` executable.
`-verbose`	CVU prints detailed output.

Usage Notes

This command does not verify the integrity of OCR contents. You must use the OCRCHECK utility to verify the contents of OCR.

Example

Verifying the integrity of OCR on the local node

To verify the integrity of OCR on the local node, run the following command:

```
cluvfy comp ocr
```

This command produces output similar to the following:

```
Verifying OCR integrity

Checking OCR integrity...

Checking the absence of a non-clustered configurationl...
All nodes free of non-clustered, local-only configurations


ASM Running check passed. ASM is running on all specified nodes

Checking OCR config file "/etc/oracle/ocr.loc"...

OCR config file "/etc/oracle/ocr.loc" check successful


Disk group for ocr location "+DATA" available on all the nodes


NOTE:
This check does not verify the integrity of the OCR contents. Execute 'ocrcheck'
as a privileged user to verify the contents of OCR.

OCR integrity check passed
```

```
Verification of OCR integrity was successful.
```

cluvfy comp ohasd

Use the `cluvfy comp ohasd` component verification command to check the integrity of the Oracle High Availability Services daemon.

Syntax

```
cluvfy comp ohasd [-n node_list] [-verbose]
```

Parameters

Table A–18 cluvfy comp ohasd Command Parameters

Parameter	Description
`-n node_list`	The comma-delimited list of nondomain qualified node names on which to conduct the verification. If you do not specify this option, then CVU checks only the local node.
`-verbose`	CVU prints detailed output.

Example

Verifying the integrity of the Oracle High Availability Services daemon on all nodes in the cluster

To verify that the Oracle High Availability Services daemon is operating correctly on all nodes in the cluster, use the following command:

```
cluvfy comp ohasd -n all -verbose
```

This command produces output similar to the following:

```
Verifying OHASD integrity

Checking OHASD integrity...
ohasd is running on node "node1"
ohasd is running on node "node2"
ohasd is running on node "node3"
ohasd is running on node "node4"

OHASD integrity check passed

Verification of OHASD integrity was successful.
```

cluvfy comp olr

Use the `cluvfy comp olr` component verification command to check the integrity of Oracle Local Registry (OLR) on the local node.

Syntax

```
cluvfy comp olr [-n node_list] [-verbose]
```

Parameters

Table A–19 cluvfy comp olr Command Parameters

Parameter	Description
`-n` *node_list*	The comma-delimited list of nondomain qualified node names on which to conduct the verification. If you do not specify this option, then CVU checks only the local node.
`-verbose`	CVU prints detailed output.

Usage Notes

This command does not verify the integrity of the OLR contents. You must use the `ocrcheck -local` command to verify the contents of OCR.

Example

Verifying the integrity of the OLR on a node

To verify the integrity of the OLR on the current node, run the following command:

```
cluvfy comp olr -verbose
```

This command produces output similar to the following:

```
Verifying OLR integrity

Checking OLR integrity...

Checking OLR config file...

OLR config file check successful


Checking OLR file attributes...

OLR file check successful

WARNING:
This check does not verify the integrity of the OLR contents. Execute ‘ocrcheck
-local’ as a privileged user to verify the contents of OLR.

OLR integrity check passed

Verification of OLR integrity was successful.
```

cluvfy comp peer

Use the `cluvfy comp peer` component verification command to check the compatibility and properties of the specified nodes against a reference node. You can check compatibility for non-default user group names and for different releases of the Oracle software. This command compares physical attributes, such as memory and swap space, and user and group values, kernel settings, and installed operating system packages.

Syntax

```
cluvfy comp peer -n node_list [-refnode node]
```

```
[-r {10.1 | 10.2 | 11.1 | 11.2 | 12.1}] [-orainv orainventory_group]
[-osdba osdba_group] [-verbose]
```

Parameters

Table A–20 cluvfy comp peer Command Parameters

Parameter	Description
`-refnode`	The node that CVU uses as a reference for checking compatibility with other nodes. If you do not specify this option, then CVU reports values for all the nodes in the node list.
`-n` *`node_list`*	The comma-delimited list of nondomain qualified node names on which to conduct the verification.
`-r {10.1 | 10.2 | 11.1 | 11.2 | 12.1}`	Specifies the software release that CVU checks as required for installation of Oracle Clusterware or Oracle RAC. If you do not specify this option, then CVU assumes Oracle Clusterware 12*c* or Oracle Database 12*c*.
`-orainv` *`orainventory_group`*	The name of the Oracle Inventory group. If you do not specify this option, then CVU uses `oinstall` as the inventory group. **Note:** This parameter is not available on Windows systems.
`-osdba` *`osdba_group`*	The name of the OSDBA group. If you do not specify this option, then CVU uses `dba` as the OSDBA group. **Note:** This parameter is not available on Windows systems.
`-verbose`	CVU prints detailed output.

Usage Notes

Peer comparison with the `-refnode` option compares the system properties of other nodes against the reference node. If the value of the other node is not greater than the value for that of the reference node, then CVU flags that comparison as a deviation from the reference node. If a group or user exists on neither the reference node nor on the other node, then CVU reports a match to indicate that there is no deviation from the reference node. Similarly, CVU reports as mismatched a comparison with a node that has less total memory than the reference node.

Example

Comparing the configuration of select cluster nodes

The following command lists the values of several preselected properties on different nodes from Oracle Database 12*c*:

```
cluvfy comp peer -n node1,node2,node4,node7 -verbose
```

cluvfy comp scan

Use the `cluvfy comp scan` component verification command to check the Single Client Access Name (SCAN) configuration.

Syntax

```
cluvfy comp scan [-verbose]
```

If you include the `-verbose` option, then CVU prints detailed output.

Example

Verifying the SCAN configuration

To verify that the SCAN and SCAN listeners are configured and operational on all nodes in the cluster, use the following command:

```
cluvfy comp scan
```

This command produces output similar to the following:

```
Verifying scan

Checking Single Client Access Name (SCAN)...

Checking TCP connectivity to SCAN Listeners...
TCP connectivity to SCAN Listeners exists on all cluster nodes

Checking name resolution setup for "node1.example.com"...

Verification of SCAN VIP and Listener setup passed

Verification of scan was successful.
```

cluvfy comp software

Use the `cluvfy comp software` component verification command to check the files and attributes installed with the Oracle software.

Syntax

```
cluvfy comp software [-n node_list] [-d oracle_home]
  [-r {10.1 | 10.2 | 11.1 | 11.2 | 12.1}] [-allfiles] [-verbose]
```

Parameters

Table A–21 cluvfy comp software Command Parameters

Parameter	Description
`-n` *node_list*	The comma-delimited list of nondomain qualified node names on which to conduct the verification. If you do not specify this option, then CVU checks only the local node.
`-d` *oracle_home*	The directory where the Oracle Database software is installed. If you do not specify this option, then the files installed in the Grid home are verified.
`-r {10.1 | 10.2 | 11.1 | 11.2 | 12.1}`	Specifies the software release that CVU checks as required for installation of Oracle Clusterware or Oracle RAC. If you do not specify this option, then CVU assumes Oracle Clusterware 12*c* or Oracle Database 12*c*.
`-allfiles`	If you specify this parameter, then CVU checks the attributes of all files of the specified Oracle home. If you do not specify this parmater, then CVU checks the attributes of the `lib`, `jlib`, and `bin` files under the specified Oracle home.
`-verbose`	CVU prints detailed output.

Example

Verifying the software configuration on all nodes in the cluster for the Oracle Clusterware home directory

To verify that the installed files for Oracle Clusterware 12*c* are configured correctly, use a command similar to the following:

```
cluvfy comp software -n all -verbose
```

This command produces output similar to the following:

```
Verifying software

Check: Software

 1021 files verified

Software check passed

Verification of software was successful.
```

cluvfy comp space

Use the `cluvfy comp space` component verification command to check for free disk space at the location you specify in the `-l` option on all the specified nodes.

Syntax

```
cluvfy comp space [-n node_list] -l storage_location
  -z disk_space {B | K | M | G} [-verbose]
```

Parameters

Table A–22 cluvfy comp space Command Parameters

Parameter	Description
`-n` *`node_list`*	The comma-delimited list of nondomain qualified node names on which to conduct the verification. If you do not specify this option, then CVU checks only the local node.
`-l` *`storage_location`*	The directory path to the storage location to check
`-z` *`disk_space`* `{B|K|M|G}`	The required disk space, in units of bytes (B), kilobytes (K), megabytes (M), or gigabytes (G). There should be no space between the numeric value and the byte indicator, for example, 2G. Use only whole numbers.
`-verbose`	CVU prints detailed output.

Usage Notes

The `space` component does not support block or raw devices.

> **See Also:** The Oracle Certification site on My Oracle Support for the most current information about certified storage options:
>
> https://support.oracle.com

Examples

Verifying the availability of free space on all nodes

You can verify that each node has 5 GB of free space in the /home/dbadmin/products directory by running the following command:

```
cluvfy comp space -n all -l /home/dbadmin/products -z 5G -verbose
```

cluvfy comp ssa

Use the cluvfy comp ssa component verification command to discover and check the sharing of the specified storage locations. CVU checks sharing for nodes in the node list.

Syntax

```
cluvfy comp ssa [-n node_list | -flex -hub hub_list [-leaf leaf_list]]
  [-s storage_path_list] [-t {software | data | ocr_vdisk}]  [-asm] [-asmdev asm_
device_list]
  [-r {10.1 | 10.2 | 11.1 | 11.2 | 12.1}] [-verbose]
```

Parameters

Table A–23 cluvfy comp ssa Command Parameters

Parameter	Description
-n *node_list* | -flex -hub *hub_list* -leaf *leaf_list*	The comma-delimited list of nondomain qualified node names on which to conduct the verification. Optionally, you can check sharing of storage locations on Hub and Leaf Nodes by specifying either -hub, followed by a comma-delimited list of Hub Node names or -leaf, followed by a comma-delimited list of Leaf Node names, or both. If you do not specify any of these options, then CVU checks only the local node.
-s *storage_path_list*	A comma-delimited list of storage paths, for example, /dev/sda,/dev/sdb. If you do not specify the -s option, then CVU discovers supported storage types and checks sharing for each of them.
-t {software | data | ocr_vdisk}	The type of Oracle files (either Oracle Grid Infrastructure binaries or Oracle Database binaries) that will be stored on the storage device. If you do not specify -t, then CVU discovers or checks the data file type.
-asm	Specify this parameter to discover all storage suitable for use by Oracle ASM.
-asmdev *asm_device_list*	A comma-delimited list of Oracle ASM devices for which you want to check sharing of storage locations. If the list contains shell metacharacters, then enclose the list in double quotation marks ("").
-r {10.1 | 10.2 | 11.1 | 11.2 | 12.1}	Optionally, you can specify the release number of the product for which you are running the verification. If you do not specify -r, then CVU runs the verification for 12.1.
-verbose	CVU prints detailed output.

Usage Notes

- The current release of `cluvfy` has the following limitations on Linux regarding shared storage accessibility check.
  - Currently NAS storage and OCFS2 (version 1.2.1 or higher) are supported.

    See Also: *Oracle Grid Infrastructure Installation Guide* for more information about NAS mount options

  - When checking sharing on NAS, `cluvfy` commands require that you have write permission on the specified path. If the `cluvfy` user does not have write permission, `cluvfy` reports the path as `not shared`.
- To perform discovery and shared storage accessibility checks for SCSI disks on Linux systems, CVU requires the CVUQDISK package. If you attempt to use CVU and the CVUQDISK package is not installed on all of the nodes in your Oracle RAC environment, then CVU responds with an error. See "Shared Disk Discovery on Red Hat Linux" on page A-43 for information about how to install the CVUQDISK package.

Examples

Example 1: Discovering All of the Available Shared Storage Systems on Your System

To discover all of the shared storage systems available on your system:

```
cluvfy comp ssa -n all -verbose
```

Example 2: Using the Oracle ASM Discovery String Check

To verify the Oracle ASM discovery string:

```
cluvfy comp ssa -n node1,node2 -asm -asmdev "/dev/xda*"
```

Example 3: Verifying the Accessibility of a Specific Storage Location

You can verify the accessibility of specific storage locations, such as an Oracle ASM disk group called `OCR13`, for storing data files for all the cluster nodes by running a command similar to the following:

```
cluvfy comp ssa -n all -s OCR13
```

This command produces output similar to the following:

```
Verifying shared storage acessibility

Checking shared storage accessibility...

"OCR13" is shared


Shared storage check was successful on nodes "node1,node2,node3,node4"

Verification of shared storage accessibility was successful.
```

cluvfy comp sys

Use the `cluvfy comp sys` component verification command to check that the minimum system requirements are met for the specified product on all the specified nodes.

Parameters

Table A–24 cluvfy comp sys Command Parameters

Parameter	Description
`-n` *node_list*	The comma-delimited list of nondomain qualified node names on which to conduct the verification. If you do not specify this option, then CVU checks only the local node.
`-p {crs | ha | database}`	Specifies whether CVU checks the system requirements for Oracle Clusterware, Oracle Restart (HA), or Oracle RAC. **Note:** Oracle does not support Oracle Restart for Oracle Database 10*g*. If you use the `-p ha` option with `-r 10.1 | 10.2`, then the check will return an error.
`-r {10.1 | 10.2 | 11.1 | 11.2 | 12.1}`	Specifies the Oracle Database release that CVU checks as required for installation of Oracle Clusterware or Oracle RAC. If you do not specify this option, then CVU assumes Oracle Database 12*c*.
`-osdba` *osdba_group*	The name of the OSDBA group. If you do not specify this option, then CVU uses `dba` as the OSDBA group.
`-orainv` *orainventory_group*	The name of the Oracle Inventory group. If you do not specify this option, then CVU uses `oinstall` as the inventory group.
`-fixup`	Specifies that if the verification fails, then CVU performs fixup operations, if feasible.
`-fixupnoexec`	Specifies that if verification fails, then CVU generates the fixup data and displays the instructions for manual execution of the generated fixups.
`-method {sudo -user` *user_name* `[-location` *directory_path*`] | root}`	Specify whether the privilege delegation method is `sudo` or `root`, for root user access. If you specify `sudo`, then you must specify the user name to access all the nodes with root privileges and, optionally, provide the full file system path for the `sudo` executable.
`-verbose`	CVU prints detailed output.

Syntax

```
cluvfy comp sys [-n node_list | -flex -hub hub_list [-leaf leaf_list]]
  -p {crs | ha | database}  [-r {10.1 | 10.2 | 11.1 | 11.2 | 12.1}] [-osdba osdba_
group]
  [-orainv orainventory_group] [-fixup] [-fixupnoexec] [-method {sudo -user user_
name
  [-location directory_path] | root}] [-verbose]
```

Examples

Verifying the system requirements for installing Oracle Clusterware

To verify the system requirements for installing Oracle Clusterware 12*c* on the cluster nodes `node1`,`node2` and `node3`, run the following command:

```
cluvfy comp sys -n node1,node2,node3 -p crs -verbose
```

cluvfy comp vdisk

Use the `cluvfy comp vdisk` component verification command to check the voting files configuration and the `udev` settings for the voting files on all the specified nodes.

> **See Also:** *Oracle Grid Infrastructure Installation Guide for Linux* for more information about `udev` settings

Syntax

```
cluvfy comp vdisk [-n node_list] [-verbose]
```

Parameters

Table A–25 cluvfy comp vdisk Command Parameters

Parameter	Description
`-n node_list`	The comma-delimited list of nondomain qualified node names on which to conduct the verification. If you do not specify this option, then CVU checks only the local node.
`-verbose`	CVU prints detailed output.

cluvfy stage [-pre | -post] acfscfg

Use the `cluvfy stage -pre acfscfg` command to verify your cluster nodes are set up correctly before configuring Oracle Automatic Storage Management Cluster File System (Oracle ACFS).

Use the `cluvfy stage -post acfscfg` to check an existing cluster after you configure Oracle ACFS.

Syntax

```
cluvfy stage -pre acfscfg -n node_list [-asmdev asm_device_list] [-verbose]

cluvfy stage -post acfscfg -n node_list [-verbose]
```

Parameters

Table A–26 cluvfy stage [-pre | -post] acfscfg Command Parameters

Parameter	Description
`-n node_list`	The comma-delimited list of nondomain qualified node names on which to conduct the verification.
`-asmdev asm_device_list`	The list of devices you plan for Oracle ASM to use. If you do not specify this option, then CVU uses an internal operating system-dependent value, for example, `/dev/raw/*` on Linux systems.
`-verbose`	CVU prints detailed output.

cluvfy stage [-pre | -post] cfs

Use the `cluvfy stage -pre cfs` stage verification command to verify your cluster nodes are set up correctly before setting up OCFS2.

Use the `cluvfy stage -post cfs` stage verification command to perform the appropriate checks on the specified nodes after setting up OCFS2.

> **See Also:** *Oracle Grid Infrastructure Installation Guide* for your platform for a list of supported shared storage types

Syntax

```
cluvfy stage -pre cfs -n node_list -s storageID_list [-verbose]

cluvfy stage -post cfs -n node_list -f file_system [-verbose]
```

Parameters

Table A–27 cluvfy stage [-pre | -post] cfs Command Parameters

Parameter	Description
`-n node_list`	The comma-delimited list of nondomain qualified node names on which to conduct the verification.
`-s storageID_list`	The comma-delimited list of storage locations to check.
`-verbose`	CVU prints detailed output.

Example

To check that a shared device is configured correctly before setting up OCFS2, use a command similar to the following, where you replace /dev/sdd5 with the name of the shared device on your system:

```
cluvfy stage -pre cfs -n node1,node2,node3,node4 -s /dev/sdd5
```

cluvfy stage [-pre | -post] crsinst

Use the `cluvfy stage -pre crsinst` command with either the `-file`, `-n`, `-flex`, or `-upgrade` parameters to check the specified nodes before installing or upgrading Oracle Clusterware.

Use the `cluvfy stage -post crsinst` command to check the specified nodes after installing Oracle Clusterware.

Syntax

```
cluvfy stage -pre crsinst -file config_file [-fixup] [-fixupnoexec]
  [-method {sudo -user user_name [-location directory_path] | root}] [-verbose]

cluvfy stage -pre crsinst -n node_list | -flex -hub hub_list [-leaf leaf_list]
  [-r {10.1 | 10.2 | 11.1 | 11.2 | 12.1}] [-c ocr_location_list] [-q voting_disk_
list]
  [-osdba osdba_group] [-orainv orainventory_group] [-asm [-presence {local |
flex}
  | -asmcredentials client_data_file] [-asmgrp asmadmin_group] [-asmdev asm_
device_list]]
  [-crshome Grid_home] [-fixup] [-fixupnoexec]
  [-method {sudo -user user_name [-location directory_path] | root}]
  [-networks network_list] [-dhcp -clustername cluster_name [-dhcpport dhcp_port]]
  [-verbose]

crsinst -upgrade [-rolling] [-src_crshome src_crshome] -dest_crshome dest_crshome
   -dest_version dest_version [-fixup] [-fixupnoexec] [-method {sudo -user user_
name
  [-location directory_path] | root}] [-verbose]

cluvfy stage -post crsinst -n node_list
```

```
[-method {sudo -user user_name [-location directory_path] | root}] [-verbose]
```

Parameters

Table A–28 cluvfy stage [-pre | -post] crsinst Command Parameters

Parameter	Description
`-file` *config_file*	Specify the root script configuration file containing Oracle installation variables.
`-n` *node_list*	The comma-delimited list of nondomain qualified node names on which to conduct the verification.
`-flex -hub` *hub_list* `[-leaf` *leaf_list*`]`	Alternative to the `-n` parameter, specify a comma-delimited list of Hub Node names on which to conduct checks. Optionally, you can specify a comma-delimited list of Leaf Nodes.
`-r {10.1 | 10.2 | 11.1 | 11.2 | 12.1}`	Specifies the Oracle Clusterware release that CVU checks as required for installation of Oracle Clusterware. If you do not specify this option, then CVU assumes Oracle Clusterware 12*c*.
`-c` *ocr_location_list*	Specify a comma-delimited list of directory paths for OCR locations or files that CVU checks for availability to all nodes. If you do not specify this option, then the OCR locations are not checked.
`-q` *voting_disk_list*	Specify a comma-delimited list of directory paths for voting files that CVU checks for availability to all nodes. If you do not specify this option, then the voting file locations are not checked
`-osdba` *osdba_group*	Specify the name of the OSDBA group. If you do not specify this option, then CVU uses `dba` as the OSDBA group.
`-orainv` *orainventory_group*	Specify the name of the Oracle Inventory group. If you do not specify this option, then CVU uses `oinstall` as the inventory group.

`-asm [-presence {local | flex} |`
`-asmcredentials client_ data_file]`
	This parameter indicates that Oracle ASM is used for storing the Oracle Clusterware files.
	Specify the Oracle ASM presence, either `LOCAL` or `FLEX`, on this Oracle Clusterware installation. Optionally, for an Oracle ASM client, specify the path to an Oracle ASM client credential file.

`-asmgrp` *asmadmin_group*	Specify the name of the OSASM group. If you do not specify this option, then CVU uses the same group as the Oracle Inventory group.
`-asmdev` *asm_device_list*	Specify a list of devices you plan for Oracle ASM to use that CVU checks for availability to all nodes. If you do not specify this option, then CVU uses an internal operating system-dependent value.
`-crshome` *Grid_home*	Specify the location of the Oracle Grid Infrastructure or Oracle Clusterware home directory. If you specify this option, then the supplied file system location is checked for sufficient free space for an Oracle Clusterware installation.

Table A–28 (Cont.) cluvfy stage [-pre | -post] crsinst Command Parameters

Parameter	Description
`-networks [network_list]`	Checks the network parameters of a slash ("/")-delimited list of networks in the form of `"if_name" [:subnet_id [:public | :cluster_interconnect]]`. ■ You can use the asterisk (`*`) wildcard character when you specify the network interface name (*if_name*), such as `eth*`, to match interfaces. ■ Specify a subnet number for the network interface for the *subnet_id* variable and choose the type of network interface.
`-dhcp -clustername cluster_name [-dhcpport dhcp_port]`	Specify the name of the cluster. Optionally, you can specify the port to which the DHCP packets will be sent. The default value for this port is 67.
`-rolling`	Specify this if the upgrade you want to perform is a rolling upgrade.
`-src_crshome src_crshome`	Specify the location of the source Grid home.
`-dest_crshome dest_crshome`	Specify the location of the destination Grid home.
`-dest_version dest_version`	Specify the version to which you are upgrading, including any patchset, such as 11.2.0.1.0 or 11.2.0.2.0.
`-fixup`	Specifies that if the verification fails, then CVU performs fixup operations, if feasible.
`-fixupnoexec`	Specifies that if verification fails, then CVU generates the fixup data and displays the instructions for manual execution of the generated fixups.
`-method {sudo -user user_name [-location dir_path] | root}`	Specify whether the privilege delegation method is `sudo` or `root`, for root user access. If you specify `sudo`, then you must specify the user name to access all the nodes with root privileges and, optionally, provide the full file system path for the `sudo` executable.
`-verbose`	CVU prints detailed output.

Usage Notes

- To perform checks for a new installation, specify either the `-file` or `-n` parameters, and use `-upgrade` for performing checks for upgrading to another version.
- CVU performs additional checks on OCR and voting files if you specify the `-c` and `-q` options with the `-n` parameter.

cluvfy stage -pre dbcfg

Use the `cluvfy stage -pre dbcfg` command to check the specified nodes before configuring an Oracle RAC database to verify whether your system meets all of the criteria for creating a database or for making a database configuration change.

Syntax

On Linux and UNIX platforms:

```
cluvfy stage -pre dbcfg -n node_list -d Oracle_home [-fixup] [-fixupnoexec]
  [-method {sudo -user user_name [-location directory_path] | root}]
  [-servicepwd] [-verbose]
```

On Windows platforms:

```
cluvfy stage -pre dbcfg -n node_list -d oracle_homew [-fixup] [-fixupnoexec]
  [-verbose] [-servicepwd]
```

Parameters

Table A–29 cluvfy stage -pre dbcfg Command Parameters

Parameter	Description
`-n node_list`	The comma-delimited list of nondomain qualified node names on which to conduct the verification.
`-d Oracle_home`	The location of the Oracle home directory for the database that is being checked.
`-fixup`	Specifies that if the verification fails, then CVU performs fixup operations, if feasible.
`-fixupnoexec`	Specifies that if verification fails, then CVU generates the fixup data and displays the instructions for manual execution of the generated fixups.
`-method {sudo -user user_name [-location dir_path] | root}`	Specify whether the privilege delegation method is `sudo` or `root`, for root user access. If you specify `sudo`, then you must specify the user name to access all the nodes with root privileges and, optionally, provide the full file system path for the `sudo` executable.
`-servicepwd`	If you specify this option, then CVU performs checks similar to those performed by the `cluvfy stage -pre dbinst` command when you specify the `-seviceuser` option. CVU determines the user name from the registry, then prompts you for the password for the service user, even if the wallet exists. CVU checks the password you enter against the password stored in the wallet. If the service password is not in the wallet or you did not specify the `-servicepwd` option, then CVU does not check the user name and password. **Note:** This parameter only applies to Windows.
`-verbose`	CVU prints detailed output.

cluvfy stage -pre dbinst

Use the `cluvfy stage -pre dbinst` command to check the specified nodes before installing or creating an Oracle RAC database to verify that your system meets all of the criteria for installing or creating an Oracle RAC database.

Syntax

On Linux and UNIX platforms:

```
cluvfy stage -pre dbinst -n <node_list> [-r {10.1 | 10.2 | 11.1 | 11.2 | 12.1}]
  [-osdba osdba_group] [-osbackup osbackup_group] [-osdg osdg_group]
  [-oskm oskm_group] [-d oracle_home] [-fixup] [-fixupnoexec]
  [-method {sudo -user user_name [-location directory_path] | root}]
  [-verbose]

cluvfy stage pre dbinst -upgrade -src_dbhome src_dbhome [-dbname dbname-list]
  -dest_dbhome dest_dbhome -dest_version dest_version
  [-fixup] [-fixupnoexec] [-method sudo -user user_name [-location directory_path]
```

On Windows platforms:

```
cluvfy stage -pre dbinst -n node_list [-r {10.1 | 10.2 | 11.1 | 11.2 | 12.1}]
  [-d Oracle_home] [-fixup] [-fixupnoexec] [-serviceuser user_name
[-servicepasswd]]]
  [-verbose]

cluvfy stage -pre dbinst -upgrade -src_dbhome src_dbhome [-dbname dbname-list
  -dest_dbhome dest_dbhome -dest_version dest_version [-fixup] [-fixupnoexec]
  [-verbose]]
```

Parameters

Table A–30 cluvfy stage -pre dbinst Command Parameters

Parameter	Description
`-n` *`node_list`*	The comma-delimited list of nondomain qualified node names on which to conduct the verification.
`-r {10.1 | 10.2 | 11.1 | 11.2 | 12.1}`	Specifies the Oracle Database release that CVU checks as required for installation of Oracle RAC. If you do not specify this option, then CVU assumes Oracle Database 12*c*.
`-osdba` *`osdba_group`*	The name of the OSDBA group. If you do not specify this option, then CVU uses `dba` as the OSDBA group.
`-d` *`Oracle_home`*	The location of the Oracle home directory where you are installing Oracle RAC and creating the Oracle RAC database. If you specify this option, then the specified location is checked for sufficient free disk space for a database installation.
`-osbackup` *`osbackup_group`*	Specify the name of the OSBACKUP group.
`-osdg` *`osdg_group`*	Specify the name of the OSDG group.
`-oskm` *`oskm_group`*	Specify the name of the OSKM group.
`-fixup`	Specifies that if the verification fails, then CVU performs fixup operations, if feasible.
`-upgrade`	Specify this parameter to verify upgrade prerequisites.
`-src_dbhome` *`src_dbhome`*	Specify the location of the source database home from which you are upgrading.
`-dbname` *`dbname-list`*	Specify a comma-delimited list of unique names of the databases you want to upgrade.
`-dest_dbhome dest_dbhome`	Specify the location of the destination database home to which you are upgrading.
`-dest_version` *`dest_version`*	Specify the version to which you are upgrading, including any patchset, such as 11.2.0.1.0 or 11.2.0.2.0.
`-fixupnoexec`	Specifies that if verification fails, then CVU generates the fixup data and displays the instructions for manual execution of the generated fixups.

Table A–30 (Cont.) cluvfy stage -pre dbinst Command Parameters

Parameter	Description
`-method {sudo -user user_name [-location dir_path] | root}`	Specify whether the privilege delegation method is `sudo` or `root`, for root user access. If you specify `sudo`, then you must specify the user name to access all the nodes with root privileges and, optionally, provide the full file system path for the `sudo` executable.
`[-serviceuser user_name [-servicepwd]]`	If you specify this option, then CVU checks the following: ■ Whether the specified user is a domain user. If the specified user is not a domain user, then CVU returns an error and does not perform any subsequent checks on the validation of the user. **Note:** You must specify the user name of the Oracle home user. ■ Whether the specified user is an administrator on all nodes in the cluster. If the user is not an administrator on any node in the cluster, then the check passes. Otherwise, the check fails. ■ If you do not specify the `-servicepwd` option, then CVU checks whether there is a password stored in the wallet on OCR for this user. If no password exists for the specified user, then CVU continues to run. ■ If you specify the `-servicepwd` option, then CVU prompts you for the password of the specified user, even if the password exists in the wallet. **Note:** The `-serviceuser` and `-servicepwd` parameters only apply to Windows.
`-verbose`	CVU prints detailed output.

cluvfy stage [-pre | -post] hacfg

Use the `cluvfy stage -pre hacfg` command to check a local node before configuring Oracle Restart.

Use the `cluvfy stage -post hacfg` command to check the local node after configuring Oracle Restart.

Syntax

```
cluvfy stage -pre hacfg [-osdba osdba_group] [-orainv orainventory_group]
  [-fixup] [-fixupnoexec] [-method {sudo -user user_name [-location directory_
path] | root}]
[-verbose]

cluvfy stage -post hacfg [-verbose]
```

Parameters

Table A–31 cluvfy stage [-pre | -post] hacfg Command Parameters

Parameter	Description
`-osdba osdba_group`	The name of the OSDBA group. If you do not specify this option, then CVU uses `dba` as the OSDBA group.
`-orainv orainventory_group`	The name of the Oracle Inventory group. If you do not specify this option, then CVU uses `oinstall` as the inventory group.
`-fixup`	Specifies that if the verification fails, then CVU performs fixup operations, if feasible.
`-fixupnoexec`	Specifies that if verification fails, then CVU generates the fixup data and displays the instructions for manual execution of the generated fixups.
`-method {sudo -user user_name [-location dir_path] | root}`	Specify whether the privilege delegation method is `sudo` or `root`, for root user access. If you specify `sudo`, then you must specify the user name to access all the nodes with root privileges and, optionally, provide the full file system path for the `sudo` executable.
`-verbose`	CVU prints detailed output.

cluvfy stage -post hwos

Use the `cluvfy stage -post hwos` stage verification command to perform network and storage verifications on the specified nodes in the cluster before installing Oracle software. This command also checks for supported storage types and checks each one for sharing.

Syntax

```
cluvfy stage -post hwos -n node_list [-s storageID_list] [-verbose]
```

Parameters

Table A–32 cluvfy stage -post hwos Command Parameters

Parameter	Description
`-n node_list`	The comma-delimited list of nondomain qualified node names on which to conduct the verification.
`-s storageID_list`	Checks the comma-delimited list of storage locations for sharing of supported storage types. If you do not provide the `-s` option, then CVU discovers supported storage types and checks sharing for each of them.
`-verbose`	CVU prints detailed output.

cluvfy stage [-pre | -post] nodeadd

Use the `cluvfy stage -pre nodeadd` command to verify the specified nodes are configured correctly before adding them to your existing cluster, and to verify the integrity of the cluster before you add the nodes.

This command verifies that the system configuration, such as the operating system version, software patches, packages, and kernel parameters, for the nodes that you want to add, is compatible with the existing cluster nodes, and that the clusterware is successfully operating on the existing nodes. Run this command on any node of the existing cluster.

Use the `cluvfy stage -post nodeadd` command to verify that the specified nodes have been successfully added to the cluster at the network, shared storage, and clusterware levels.

Syntax

```
cluvfy stage -pre nodeadd -n node_list [-vip vip_list] | -flex [-hub hub_list
  [-vip vip_list]] [-leaf leaf_list] [-fixup] [-fixupnoexec]
  [-method {sudo -user user_name [-location directory_path] | root}] [-verbose]

cluvfy stage -post nodeadd -n node_list [-verbose]
```

Parameters

Table A–33 cluvfy stage [-pre | -post] nodeadd Command Parameters

Parameter	Description
`-n` *`node_list`*	A comma-delimited list of nondomain qualified node names on which to conduct the verification. These are the nodes you are adding or have added to the cluster.
`-vip` *`vip_list`*	A comma-delimited list of virtual IP addresses to be used by the new nodes.

`-flex [-hub` *`hub_list`*
`[-vip` *`vip_list`*`]]`
	Specify `-flex` if you are adding a node to an Oracle Flex Cluster. Optionally, you can specify a comma-delimited list of non-domain qualified node names that you want to add to the cluster as Hub Nodes.
	Additionally, you can specify a comma-delimited list of virtual IP addresses that will be applied to the list of Hub Nodes you specify.

`-leaf` *`leaf_list`*	Optionally, you can specify a comma-delimited list of non-domain qualified node names that you want to add to the cluster as Leaf Nodes.
`-fixup`	Specifies that if the verification fails, then CVU performs fixup operations, if feasible.
`-fixupnoexec`	Specifies that if verification fails, then CVU generates the fixup data and displays the instructions for manual execution of the generated fixups.

`-method {sudo -user` *`user_name`*
`[-location` *`dir_path`*`] | root}`
	Specify whether the privilege delegation method is `sudo` or `root`, for root user access. If you specify `sudo`, then you must specify the user name to access all the nodes with root privileges and, optionally, provide the full file system path for the `sudo` executable.

`-verbose`	CVU prints detailed output.

cluvfy stage -post nodedel

Use the `cluvfy stage -post nodedel` command to verify that specific nodes have been successfully deleted from a cluster. Typically, this command verifies that the node-specific interface configuration details have been removed, the nodes are no longer a part of cluster configuration, and proper Oracle ASM cleanup has been performed.

Syntax

```
cluvfy stage -post nodedel -n node_list [-verbose]
```

Parameters

Table A–34 cluvfy stage -post nodedel Command Parameters

Parameter	Description
`-n` *`node_list`*	The comma-delimited list of nondomain qualified node names that have been removed from the cluster.
`-verbose`	CVU prints detailed output.

Usage Notes

If the `cluvfy stage -post nodedel` check fails, then repeat the node deletion procedure.

See Also: Chapter 7, "Adding and Deleting Cluster Nodes"

Troubleshooting and Diagnostic Output for CVU

This section describes the following troubleshooting topics for CVU:

- Enabling Tracing
- Known Issues for the Cluster Verification Utility

Enabling Tracing

CVU generates trace files unless you disable tracing. You can disable tracing by setting the `SRVM_TRACE` environment variable to `false` or `FALSE`. For example, in `tcsh` an entry such as `setenv SRVM_TRACE FALSE` disables tracing.

The CVU trace files are created in the `ORACLE_BASE/crsdata/`*`host_name`*`/cvu` directory by default. Oracle Database automatically rotates the log files and the most recently created log file has the name `cvutrace.log.0`. You should remove unwanted log files or archive them to reclaim disk place if needed.

Oracle Clusterware stores log files that CVU generates when it runs periodically in the `ORACLE_BASE/crsdata/`*`host_name`*`/cvu/cvutrc` directory.

To use a non-default location for the trace files, set the `CV_TRACELOC` environment variable to the absolute path of the desired trace directory.

Known Issues for the Cluster Verification Utility

This section describes the following known limitations for Cluster Verification Utility (CVU):

- Database Versions Supported by Cluster Verification Utility
- Linux Shared Storage Accessibility (ssa) Check Reports Limitations
- Shared Disk Discovery on Red Hat Linux

Database Versions Supported by Cluster Verification Utility

The current CVU release supports only Oracle Database 10*g* or higher, Oracle RAC, and Oracle Clusterware; CVU is not backward compatible. CVU cannot check or verify Oracle Database products for releases before Oracle Database 10*g*.

Linux Shared Storage Accessibility (ssa) Check Reports Limitations

The current release of cluvfy has the following limitations on Linux regarding shared storage accessibility check.

- OCFS2 (version 1.2.1 or higher) is supported.
- For sharedness checks on NAS, cluvfy commands require you to have write permission on the specified path. If the user running the cluvfy command does not have write permission, then cluvfy reports the path as not shared.

Shared Disk Discovery on Red Hat Linux

To perform discovery and shared storage accessibility checks for SCSI disks on Red Hat Linux 5.0 (or higher) and Oracle Linux 5.0 (or higher), and SUSE Linux Enterprise Server, CVU requires the CVUQDISK package. If you attempt to use CVU and the CVUQDISK package is not installed on all of the nodes in your Oracle RAC environment, then CVU responds with an error.

Perform the following procedure to install the CVUQDISK package:

1. Login as the root user.
2. Copy the package, cvuqdisk-1.0.9-1.rpm (or higher version) to a local directory. You can find this rpm in the rpm subdirectory of the top-most directory in the Oracle Clusterware installation media. For example, you can find cvuqdisk-1.0.9-1.rpm in the directory /*mountpoint*/clusterware/rpm/ where *mountpoint* is the mount point for the disk on which the directory is located.

   ```
   # cp /mount_point/clusterware/rpm/cvuqdisk-1.0.9-1.rpm /u01/oradba
   ```

3. Set the CVUQDISK_GRP environment variable to the operating system group that should own the CVUQDISK package binaries. If CVUQDISK_GRP is not set, then, by default, the oinstall group is the owner's group.

   ```
   # set CVUQDISK_GRP=oinstall
   ```

4. Determine whether previous versions of the CVUQDISK package are installed by running the command rpm -q cvuqdisk. If you find previous versions of the CVUQDISK package, then remove them by running the command rpm -e cvuqdisk *previous_version* where *previous_version* is the identifier of the previous CVUQDISK version, as shown in the following example:

   ```
   # rpm -q cvuqdisk
   cvuqdisk-1.0.2-1
   # rpm -e cvuqdisk-1.0.2-1
   ```

5. Install the latest CVUQDISK package by running the command rpm -iv cvuqdisk-1.0.9-1.rpm.

   ```
   # cd /u01/oradba
   # rpm -iv cvuqdisk-1.0.9-1.rpm
   ```

B

Oracle Clusterware Resource Reference

This appendix is a reference for Oracle Clusterware resources. This appendix includes descriptions and usage examples of resource attributes and detailed descriptions and examples of resource attribute action scripts. This appendix includes the following topics:

- Resource Attributes
- Examples of Action Scripts for Third-party Applications

Resource Attributes

This section lists and describes attributes used when you register applications as resources in Oracle Clusterware. Use these attributes with the `crsctl add resource` command, as follows:

```
$ crsctl add resource resource_name -type resource_type
{[-attr "attribute_name='attribute_value', attribute_name='attribute_value'
, ..."] | [-file file_name]}
```

List attribute-value pairs in a comma-delimited list after the `-attr` flag and enclose the value of each attribute in single quotation marks (`''`). Some resource attributes you cannot configure and are read only.

Alternatively, you can create a text file that contains the attribute-value pairs. For example:

```
PLACEMENT=favored
HOSTING_MEMBERS=node1 node2 node3
RESTART_ATTEMPTS@CARDINALITYID(1)=0
RESTART_ATTEMPTS@CARDINALITYID(2)=0
FAILURE_THRESHOLD@CARDINALITYID(1)=2
FAILURE_THRESHOLD@CARDINALITYID(2)=4
FAILURE_INTERVAL@CARDINALITYID(1)=300
FAILURE_INTERVAL@CARDINALITYID(2)=500
CHECK_INTERVAL=2
CARDINALITY=2
```

Note: The length limit for these attributes is 254 characters.

This section includes the following topics:

- Configurable Resource Attributes
- Read-Only Resource Attributes

- Deprecated Resource Attributes

Configurable Resource Attributes

This section describes the following resource attributes that you can configure when registering an application as a resource in Oracle Clusterware:

> **Note:** Values for all attributes must be in lowercase. Attribute names must be in all uppercase letters.

- ACL
- ACTION_SCRIPT
- ACTION_TIMEOUT
- ACTIONS
- ACTIVE_PLACEMENT
- AGENT_FILENAME
- ALERT_TEMPLATE
- AUTO_START
- CARDINALITY
- CHECK_INTERVAL
- CHECK_TIMEOUT
- CLEAN_TIMEOUT
- DELETE_TIMEOUT
- DESCRIPTION
- ENABLED
- FAILURE_INTERVAL
- FAILURE_THRESHOLD
- HOSTING_MEMBERS
- INSTANCE_FAILOVER
- INTERMEDIATE_TIMEOUT
- LOAD
- MODIFY_TIMEOUT
- NAME
- OFFLINE_CHECK_INTERVAL
- PLACEMENT
- RELOCATE_BY_DEPENDENCY
- RESTART_ATTEMPTS
- SCRIPT_TIMEOUT
- SERVER_CATEGORY
- SERVER_POOLS

- START_CONCURRENCY
- START_DEPENDENCIES
- START_TIMEOUT
- STOP_CONCURRENCY
- STOP_DEPENDENCIES
- STOP_TIMEOUT
- UPTIME_THRESHOLD
- USER_WORKLOAD
- USE_STICKINESS

ACL

Defines the owner of a resource and the access privileges granted to various operating system users and groups. The resource owner defines the operating system user of the owner and its privileges. You configure this optional attribute when you create a resource. If you do not configure this attribute, then the value is based on the identity of the process creating the resource. You can change the value of the attribute if such a change is allowed based on the existing privileges of the resource.

> **Note:** All operating system user names and user groups, including `owner`, `pgrp`, `user`, and `group`, *must* be registered on *all* servers in the cluster.

In the string:

- `owner`: The operating system user that owns a resource and the user under which the action script or application-specific agent runs, followed by the privileges of the owner.
- `pgrp`: The operating system group that is the primary group of the owner of a resource, followed by the privileges of members of the primary group.
- `other`: Operating system users that are neither the owner nor member of the primary group
- `r`: The read option, which gives the ability to only see a resource, its state, and configuration
- `w`: The write option, which gives the ability to modify a resource's attributes and to delete the resource
- `x`: The execute option, which gives the ability to start, stop, and relocate a resource

By default, the identity of the client that creates a resource is the `owner`. Also by default, `root`, and the user specified in `owner` have full privileges. You can grant required operating system users and operating system groups their privileges by adding the following lines to the `ACL` attribute:

```
user:user_name:rwx
group:group_name:rwx
```

Usage Example

```
ACL=owner:user_1:rwx,pgrp:osdba:rwx,other::r-
```

In the preceding example, the owner of the resource is user_1, whose primary group is osdba. The user, user_1, has all privileges, as does the osdba group, while other users can only view the resource.

ACTION_SCRIPT

An absolute file name that includes the path and file name of an action script. The agent specified in the AGENT_FILENAME attribute calls the script specified in the ACTION_SCRIPT attribute.

Usage Example

```
ACTION_SCRIPT=fully_qualified_path_to_action_script
```

ACTION_TIMEOUT

A single timeout value, in seconds, for all supported actions that Oracle Clusterware can perform on a resource.

Usage Example

```
ACTION_TIMEOUT=30
```

ACTIONS

The ACTIONS attribute declares a table of names that lists the actions that Oracle Clusterware can perform on a resource and the permissions that correspond to the actions. The ACTIONS attribute contains a space-delimited list of action specifications, where each specification has the following format, where:

- *actionName* is the name of the action (the maximum length is 32 US7ASCII alphanumeric, case-sensitive characters)
- *userName* is an operating system user name that is enabled to perform the action
- *groupName* is an operating system group name that is enabled to perform the action

```
actionName [,user:userName | group:groupName][ ...]
```

If you do not specify a *userName* or *groupName*, then Oracle Clusterware assumes that the actions are universally accessible.

Usage Example

The following example enables multiple actions:

```
ACTIONS='action1 action2,user:user2 action3,group:group1'
```

ACTIVE_PLACEMENT

When set to 1, Oracle Clusterware uses this attribute to reevaluate the placement of a resource during addition or restart of a cluster server. For resources where PLACEMENT=favored, Oracle Clusterware may relocate running resources if the resources run on a non-favored server when a favored one joins the cluster.

Usage Example

```
ACTIVE_PLACEMENT=1
```

AGENT_FILENAME

A fully qualified file name of an agent program that a resource type uses to manage its resources. Every resource type must have an agent program to manage its resources. Resource types use agent programs by either specifying a value for this attribute or inheriting it from their base resource type. There are two script agents included with Oracle Clusterware 12*c*: `application` and `scriptagent`. Oracle Clusterware uses the `application` script agent for resources of the deprecated `application` resource type. The default value for this attribute is `scriptagent`.

> **Note:** Once the resource is created, you can no longer modify this attribute.

Usage Example

```
AGENT_FILENAME=%Grid_home%/bin/application
```

ALERT_TEMPLATE

Use to specify additional resource attributes that are to be included in resource state alert messages. You can specify the attribute as a space-delimited list of resource attributes. These attributes must be accessible from the resource type to display in alert messages.

Usage Example

```
ALERT_TEMPLATE="DESCRIPTION HOSTING_MEMBERS"
```

AUTO_START

Indicates whether Oracle Clusterware automatically starts a resource after a cluster server restart. Valid `AUTO_START` values are:

- `always`: Restarts the resource when the server restarts regardless of the state of the resource when the server stopped.
- `restore`: Restores the resource to the same state that it was in when the server stopped. Oracle Clusterware attempts to restart the resource if the value of `TARGET` was `ONLINE` before the server stopped.
- `never`: Oracle Clusterware never restarts the resource regardless of the state of the resource when the server stopped.

CARDINALITY

The number of servers on which a resource can run, simultaneously. This is the upper limit for resource cardinality.

Usage Example

```
CARDINALITY=1
```

You can also use a value such that cardinality always increases and decreases with the number of servers that are assigned to the server pool in which the resource is configured to run. The value is:

```
CARDINALITY=%CRS_SERVER_POOL_SIZE%
```

Only resources with `PLACEMENT=restricted` and that use the `SERVER_POOLS` attribute can use this value.

CHECK_INTERVAL

The time interval, in seconds, between repeated executions of the `check` action. Shorter intervals enable more frequent checks but also increase resource consumption if you use the script agent. Use an application-specific agent to reduce resource consumption.

Usage Example

```
CHECK_INTERVAL=60
```

CHECK_TIMEOUT

The maximum time, in seconds, in which a check action can run. Oracle Clusterware returns an error message if the action does not complete within the time specified. If you do not specify this attribute or if you specify 0 seconds, then Oracle Clusterware uses the value of the `SCRIPT_TIMEOUT` attribute.

Usage Example

```
CHECK_TIMEOUT=30
```

CLEAN_TIMEOUT

The maximum time, in seconds, in which a clean action can run. Oracle Clusterware returns an error message if the action does not complete within the time specified. If you do not specify a value for this attribute or you specify `0` seconds, then Oracle Clusterware uses the value of the `STOP_TIMEOUT` attribute.

Usage Example

```
CLEAN_TIMEOUT=30
```

DELETE_TIMEOUT

The maximum time, in seconds, in which a delete action can run. Oracle Clusterware returns an error message if the action does not complete within the time specified. If you do not specify a value for this attribute or you specify `0` seconds, then Oracle Clusterware uses the value of the SCRIPT_TIMEOUT attribute.

Usage Example

```
DELETE_TIMEOUT=30
```

DESCRIPTION

Enter a description of the resource you are adding.

Usage Example

```
DESCRIPTION=Apache Web server
```

ENABLED

Oracle Clusterware uses this attribute to manage the state of the resource. Oracle Clusterware does not attempt to manage a disabled (`ENABLED=0`) resource either directly or because of a dependency to another resource. A disabled resource cannot be started but it can be stopped. Oracle Clusterware does not actively monitor disabled resources, meaning that Oracle Clusterware does not check their state.

Usage Example

```
ENABLED=1
```

FAILURE_INTERVAL

The interval, in seconds, before which Oracle Clusterware stops a resource if the resource has exceeded the number of failures specified by the FAILURE_THRESHOLD attribute. If the value is zero (0), then tracking of failures is disabled.

Usage Example

```
FAILURE_INTERVAL=30
```

FAILURE_THRESHOLD

The number of failures of a resource detected within a specified FAILURE_INTERVAL for the resource before Oracle Clusterware marks the resource as unavailable and no longer monitors it. If a resource fails the specified number of times, then Oracle Clusterware stops the resource. If the value is zero (0), then tracking of failures is disabled. The maximum value is 20.

Usage Example

```
FAILURE_THRESHOLD=3
```

HOSTING_MEMBERS

A space-delimited, ordered list of cluster server names that can host a resource. This attribute is required only when using administrator management, and when the value of the PLACEMENT attribute is set to favored or restricted. When registering applications as Oracle Clusterware resources, use the SERVER_POOLS attribute, instead.

> **Note:** For resources of application type, Oracle Clusterware places servers listed in the HOSTING_MEMBERS attribute in the Generic server pool.

See Also:

- "Default Server Pools" on page 3-3 for more information about the Generic server pool
- "LOAD" on page B-8 for more information about this attribute
- "PLACEMENT" on page B-8 for more information about this attribute

To obtain a list of candidate node names, run the olsnodes command to display a list of your server names.

Usage Example

```
HOSTING_MEMBERS=server1 server2 server3
```

INSTANCE_FAILOVER

Use the INSTANCE_FAILOVER attribute for resources of type CLUSTER_RESOURCE. Using this attribute enables you to disallow the failover of resource instances from the servers on which they fail. This enables you to bind the resource to a particular server.

Set to 0 to disable instance failover.

Usage Example

```
INSTANCE_FAILOVER=1
```

INTERMEDIATE_TIMEOUT

Denotes the maximum amount of time in seconds that a resource can remain in the INTERMEDIATE state before the resource is declared as failed. The value of INTERMEDIATE_TIMEOUT must be greater than 0 to take effect.

Usage Example

```
INTERMEDIATE_TIMEOUT=60
```

LOAD

Oracle Clusterware interprets the value of this attribute along with that of the PLACEMENT attribute. When the value of PLACEMENT is balanced, the value of LOAD determines where best to place a resource. A nonnegative, numeric value that quantitatively represents how much server capacity an instance of a resource consumes relative to other resources. Oracle Clusterware attempts to place resources on servers with the least total load of running resources.

Usage Example

```
LOAD=1
```

MODIFY_TIMEOUT

The maximum time, in seconds, in which a modify action can run. Oracle Clusterware returns an error message if the action does not complete within the time specified. If you do not specify a value for this attribute or you specify 0 seconds, then Oracle Clusterware uses the value of the SCRIPT_TIMEOUT attribute.

Usage Example

```
MODIFY_TIMEOUT=30
```

NAME

A case-sensitive alphanumeric string that names the resource. Oracle recommends a naming convention that starts with an alphanumeric prefix, such as myApache, and complete the name with an identifier to describe it. A resource name can contain any platform-supported characters except the exclamation point (!) and the tilde (~). A resource name cannot begin with a period (.) nor with the string *ora*.

Usage Example

```
NAME=myApache
```

OFFLINE_CHECK_INTERVAL

Controls offline monitoring of a resource. The value represents the interval (in seconds) that Oracle Clusterware monitors a resource when its state is OFFLINE. Monitoring is disabled if the value is 0.

Usage Example

```
OFFLINE_CHECK_INTERVAL=30
```

PLACEMENT

Specifies how Oracle Clusterware selects a cluster server on which to start a resource. Valid values are balanced, favored, or restricted.

If you set the PLACEMENT attribute to favored or restricted, then you must also assign values to the SERVER_POOLS and HOSTING_MEMBERS attributes. If you set the value of the PLACEMENT attribute to balanced, then the HOSTING_MEMBERS attribute is not required.

See Also:

- "Application Placement Policies" on page 9-28 for more information about the PLACEMENT attribute
- "HOSTING_MEMBERS" on page B-7 for more information about this attribute
- "SERVER_POOLS" on page B-10 for more information about this attribute

Usage Example

```
PLACEMENT=favored
```

RELOCATE_BY_DEPENDENCY

Use to declare whether a resource will be enabled for relocation if requested to do so because of a dependency on the resource for which the relocation was requested. If 0, the resource will not be allowed to relocate because of a dependency on the resource for which relocate request was issued. The valid values are 1 or 0.

Usage Example

```
RELOCATE_BY_DEPENDENCY=1
```

RESTART_ATTEMPTS

The number of times that Oracle Clusterware attempts to restart a resource on the resource's current server before attempting to relocate it. A value of 1 indicates that Oracle Clusterware only attempts to restart the resource once on a server. A second failure causes Oracle Clusterware to attempt to relocate the resource. A value of 0 indicates that there is no attempt to restart but Oracle Clusterware always tries to fail the resource over to another server.

Usage Example

```
RESTART_ATTEMPTS=2
```

SCRIPT_TIMEOUT

The maximum time (in seconds) for an action to run. Oracle Clusterware returns an error message if the action script does not complete within the time specified. The timeout applies to all actions (start, stop, check, and clean).

Usage Example

```
SCRIPT_TIMEOUT=45
```

SERVER_CATEGORY

For local resources, the definition of a local_resource type is extended to be category-aware. In other words, you can restrict local resources to belong to a particular server category. For cluster resources, the value for the SERVER_CATEGORY attribute always functions with the value for the PLACEMENT attribute. Set SERVER_POOLS to * when PLACEMENT is restricted and SERVER_CATEGORY is used. If you set PLACEMENT to restricted, then Oracle Clusterware expects one of the following attributes to also be set:

- HOSTING_MEMBERS
- SERVER_POOLS
- SERVER_CATEGORY

For example, a resource, known as resource1, can have a policy that sets the value of PLACEMENT to be restricted, and SERVER_CATEGORY is set to HubCategory. In such a case, Oracle Clusterware would only enable resource1 to run on the servers that belong to the HubCategory.

If PLACEMENT is set to favored and if only one of HOSTING_MEMBERS, SERVER_POOLS, or SERVER_CATEGORY is set, then that value expresses a preference. If HOSTING_MEMBERS is populated and one of SERVER_POOLS or SERVER_CATEGORY is set, then the HOSTING_MEMBERS indicates placement preference and SERVER_POOLS or SERVER_CATEGORY indicates a restriction. For example, the ora.cluster.vip resource can have a policy that sets the value of PLACEMENT to favored, and SERVER_CATEGORY is set to HubCategory and HOSTING_MEMBERS is set to server_name1. In such a case, Oracle Clusterware restricts the placement of ora.cluster.vip to the servers in the HubCategory and then it prefers the server known as server_name1.

Usage Example

```
SERVER_CATEGORY=my_category
```

SERVER_POOLS

A space-delimited list of the server pools to which a particular resource can belong. If a resource can run on any server in a cluster, then use the default value, *, unless the resource is a cluster_resource type, in which case, the default value for the SERVER_POOLS attribute is empty. Only cluster administrators can specify * as the value for this attribute.

- Use the PLACEMENT attribute with the SERVER_POOLS attribute, as follows: If you set the value of the PLACEMENT attribute to either restricted or favored, then you must also provide a value for the SERVER_POOLS attribute when using policy management for the resource.
- If the value for PLACEMENT is set to balanced, then the resource only runs in the Generic and Free pools, unless SERVER_POOLS=*.

This attribute creates an affinity between a resource and one or more server pools regarding placement, and depends on the value of the PLACEMENT attribute.

See Also:

- Chapter 3, "Policy-Based Cluster and Capacity Management" for more information about server pools

  "Role-Separated Management" on page 2-1 for more information about cluster administrators
- "HOSTING_MEMBERS" on page B-7 for more information about this attribute
- "PLACEMENT" on page B-8 for more information about this attribute

Usage Example

```
SERVER_POOLS=pool1 pool2 pool3
```

START_CONCURRENCY

Describes the maximum number of start actions that can be concurrent at a time. A value of `0` means "no limit."

Usage Example

```
START_CONCURRENCY=10
```

START_DEPENDENCIES

Specifies a set of relationships that Oracle Clusterware considers when starting a resource. You can specify a space-delimited list of dependencies on several resources and resource types on which a particular resource can depend.

Syntax

```
START_DEPENDENCIES=dependency(resource_set) [dependency(resource_set)] [...]
```

In the preceding syntax example the variables are defined, as follows:

- *dependency*: Possible values are `attraction`, `dispersion`, `exclusion`, `hard`, `pullup`, and `weak`. You can specify each dependency only once, except for `pullup`, which you can specify multiple times.
- *resource_set*: A comma-delimited list of resource entities—either individual resources or resource types—enclosed in parentheses `()`, in the form of `res1[, res2[, ...]]`, upon which the resource you are configuring depends.

  Each resource entity is defined, as follows:

  ```
  [modifier1:[modifier2:]] {resource_name | type:resource_type}
  ```

  In the preceding syntax example, *resource_name* is the name of a specific resource and `type:`*resource_type* is the name of a specific resource type. The resource type must be preceded by `type` and the `type` modifier must be the last resource entity in the list.

  Optionally, you can specify modifiers to further configure resource entity dependencies. You can modify each dependency by prefixing the following modifiers to the resource entity:

  – `attraction([intermediate:]{`*resource_name* `|` `type:`*resource_type*`})`—Use the `attraction` start dependency when you want this resource to run on the same server with a particular named resource or any resource of a particular type.

    Use `intermediate` to specify that this resource is attracted to resource entities on which it depends that are in the `INTERMEDIATE` state. If not specified, then resources must be in the `ONLINE` state to attract the dependent resource.

    If you specify the `attraction` dependency on a resource type for a resource, then any resource of that particular type attracts the dependent resource.

  – `exclusion([[preempt_pre: | preempt_post:]]` *target_resource_name* `|` `type:`*target_resource_type*`])`—Use the `exclusion` start dependency to keep resources with this dependency from running on the same node.

    Use the `preempt_pre` modifier to configure the `exclusion` dependency to stop the specified target resource or resources defined by a specific resource type before starting the source resource.

Use the `preempt_post` modifier to configure the `exclusion` dependency to stop and relocate, if possible, the specified target resource or resources defined by a specific resource type after starting the source resource.

- `dispersion[:active]([intermediate:][pool:]{`*`resource_name`*` | type:`*`resource_type`*`})`—Specify the `dispersion` start dependency for a resource that you want to run on a server that is different from the named resources or resources of a particular type. Resources may still end up running on the same server, depending on availability of servers.

  Use the `active` modifier to configure the `dispersion` dependency so that Oracle Clusterware attempts to relocate the dependent resource to another server if it is collocated with another resource and another server comes online. Oracle Clusterware does not relocate resources to newly available servers unless you specify the `active` modifier.

  Use the `intermediate` modifier to specify that Oracle Clusterware can relocate the dependent resource if a resource is in either the `ONLINE` or `INTERMEDIATE` state. If not specified, then resources must be in the `ONLINE` state for dispersion of the dependent resource to occur.

  Use the `pool` modifier if you want a resource to be located in a different server pool than the target, rather than just a different server.

- `hard([intermediate:][global:][uniform:]{`*`resource_name`*` | type:`*`resource_type`*`})`—Specify a `hard` start dependency for a resource when you want the resource to start only when a particular resource or resource of a particular type starts.

  Use the `intermediate` modifier to specify that Oracle Clusterware can start this resource if a resource on which it depends is in either the `ONLINE` or `INTERMEDIATE` state. If not specified, then resources *must* be in the `ONLINE` state for Oracle Clusterware to start this resource.

  Use the `global` modifier to specify that resources are *not* required to reside on the same server as a condition to Oracle Clusterware starting this resource. If not specified, then resources must reside on the same server for Oracle Clusterware to start this resource.

  Use the `uniform` modifier to attempt to start all instances of resource B, but only one instance, at least must start to satisfy the dependency.

  If you specify the `hard` dependency on a resource type for a resource, then the resource can start if any resource of that particular type is running.

  Note: Oracle recommends that resources with `hard` start dependencies also have `pullup` start dependencies.

- `pullup[:always]([intermediate:][global:]{`*`resource_name`*` | type:`*`resource_type`*`})`—When you specify the `pullup` start dependency for a resource, then this resource starts because of named resources starting.

  Use the `always` modifier for `pullup` so that Oracle Clusterware starts this resource despite the value of its `TARGET` attribute, whether that value is `ONLINE` or `OFFLINE`. Otherwise, if you do not specify the `always` modifier, then Oracle Clusterware starts this resource only if the value of the `TARGET` attribute is `ONLINE` for this resource.

Use the `intermediate` modifier to specify that Oracle Clusterware can start this resource if a resource on which it depends is in either the `ONLINE` or `INTERMEDIATE` state. If not specified, then resources must be in the `ONLINE` state for Oracle Clusterware to start this resource.

Use the `global` modifier to specify that resources on which this resource depends are *not* required to reside on the same server as a condition to Oracle Clusterware starting this resource. If not specified, then resources on which this resource depends must reside on the same server for Oracle Clusterware to start this resource.

If you specify the `pullup` dependency on a resource type for a resource, then, when any resource of that particular type starts, Oracle Clusterware can start this resource.

Note: Oracle recommends that resources with `hard` start dependencies also have `pullup` start dependencies.

– `weak([concurrent:][global:][uniform:]{`*`resource_name`*` | type:`*`resource_type`*`})`—Specify a `weak` start dependency for a resource when you want that resource to start despite whether named resources are running, or not. An attempt to start this resource also attempts to start any resources on which this resource depends if they are not running.

Use the `concurrent` modifier to specify that Oracle Clusterware can start a dependent resource while a resource on which it depends is in the process of starting. If `concurrent` is not specified, then resources must complete startup before Oracle Clusterware can start the dependent resource.

Use the `global` modifier to specify that resources are *not* required to reside on the same server as a condition to Oracle Clusterware starting the dependent resource.

Use the `uniform` modifier to start all instances of the resource everywhere the resource can run. If you do not specify a modifier (the default), then the resource starts on the same server as the resource on which it depends.

If you specify the `weak` start dependency on a resource type for a resource, then the resource can start if any resource of that particular type is running.

See Also: "Start Dependencies" on page 9-14 for more details about start dependencies

START_TIMEOUT

The maximum time (in seconds) in which a start action can run. Oracle Clusterware returns an error message if the action does not complete within the time specified. If you do not specify a value for this attribute or you specify `0` seconds, then Oracle Clusterware uses the value of the `SCRIPT_TIMEOUT` attribute.

Usage Example

```
START_TIMEOUT=30
```

See Also: "SCRIPT_TIMEOUT" on page B-9 for more information about this attribute

STOP_CONCURRENCY

Describes the maximum number of stop actions that can be concurrent at a time. A value of 0 means "no limit."

Usage Example

```
STOP_CONCURRENCY=10
```

STOP_DEPENDENCIES

Specifies a set of relationships that Oracle Clusterware considers when stopping a resource.

Syntax

```
STOP_DEPENDENCIES=dependency(resource_set) [dependency(resource_set)] ...
```

In the preceding syntax example the variables are defined, as follows:

- *dependency*: The only possible value is hard.
- *resource_set*: A comma-delimited list, in the form of res1[, res2 [,...]], of resource entities—either individual resources or resource types—upon which the resource you are configuring depends.

  Each resource entity is defined, as follows:

  ```
  [modifier1:[modifier2:][modifier3:]] resource_name | type:resource_type
  ```

  In the preceding syntax example, *resource_name* is the name of a specific resource and type:*resource_type* is the name of a specific resource type. The resource type must be preceded by type:.

  Optionally, you can specify modifiers to further configure resource entity dependencies. You can modify each dependency by prefixing the following modifiers to the resource entity:

  hard([intermediate:][global:][shutdown:]{*resource_name* | type:*resource_type*})—Specify a hard stop dependency for a resource that you want to stop when named resources or resources of a particular resource type stop.

  Use intermediate to specify that the dependent resource can remain in an ONLINE state if a resource is in either the ONLINE or INTERMEDIATE state. If not specified, then Oracle Clusterware stops the dependent resource unless resources are in the ONLINE state.

  Use global to specify that the dependent resource remains in an ONLINE state if a resource is in an ONLINE state on any node in the cluster. If not specified, then when resources residing on the same server go offline, Oracle Clusterware stops the dependent resource.

  Use shutdown to apply this dependency when the Oracle Clusterware stack is shut down. This is a convenient way to affect the order of stopping resources when stopping the stack, without having any affect on planned or unplanned events on the individual resources. This dependency, when used with the shutdown modifier, does not go into effect if somebody stops the resource directly, but only when the stack is shut down.

  See Also: "Stop Dependencies" on page 9-18 for more details about stop dependencies

STOP_TIMEOUT

The maximum time (in seconds) in which a stop or clean action can run. Oracle Clusterware returns an error message if the action does not complete within the time specified. If you do not specify this attribute or if you specify 0 seconds, then Oracle Clusterware uses the value of the SCRIPT_TIMEOUT attribute.

Usage Example

```
STOP_TIMEOUT=30
```

See Also:

- "SCRIPT_TIMEOUT" on page B-9 for more information about this attribute
- "START_TIMEOUT" on page B-13 for more information about this attribute

UPTIME_THRESHOLD

The value for UPTIME_THRESHOLD represents the length of time that a resource must be up before Oracle Clusterware considers the resource to be stable. By setting a value for the UPTIME_THRESHOLD attribute, you can indicate the stability of a resource.

Enter values for this attribute as a number followed by a letter that represents seconds (s), minutes (m), hours (h), days (d), or weeks (w). For example, a value of 7h represents an uptime threshold of seven hours.

After the time period you specify for UPTIME_THRESHOLD elapses, Oracle Clusterware resets the value for RESTART_COUNT to 0 at the next resource state change event, such as stop, start, relocate, or failure. Oracle Clusterware can alert you when the value for RESTART_COUNT reaches the value that you set for RESTART_ATTEMPTS. The counter is effectively reset the next time the resource fails or restarts. The threshold represents the amount of time that restarts are to be counted and discarded. If the resource fails after the threshold, it will still restart.

Note: Oracle Clusterware writes an alert to the clusterware alert log file when the value for RESTART_COUNT reaches the value that you set for RESTART_ATTEMPTS.

See Also:

- "RESTART_ATTEMPTS" on page B-9 for more information about this attribute
- "RESTART_COUNT" on page B-17 for more information about this attribute

USER_WORKLOAD

Use to indicate whether a resource is a workload generating resource for what-if analysis. Possible values are yes or no.

Usage Example

```
USER_WORKLOAD=yes
```

USE_STICKINESS

Use to indicate that a resource should run where it last ran, if possible, and to not permit load-balancing that would otherwise apply. If set to 1, Oracle Clusterware attempts to start the resource where it last ran. Enabling USE_STICKINESS also disables load-balancing. The default value is 0. Possible values are 0 and 1.

Usage Example

```
USE_STICKINESS=1
```

Read-Only Resource Attributes

You can view these attributes when you run the crsctl status resource command on a particular resource. Oracle Clusterware sets these attributes when you register resources.

- ACTION_FAILURE_EVENT_TEMPLATE
- INSTANCE_COUNT
- INTERNAL_STATE
- LAST_SERVER
- LAST_STATE_CHANGE
- PROFILE_CHANGE_EVENT_TEMPLATE
- RESTART_COUNT
- STATE
- STATE_CHANGE_EVENT_TEMPLATE
- STATE_DETAILS
- TARGET
- TARGET_SERVER
- TYPE

ACTION_FAILURE_EVENT_TEMPLATE

This is an internally-managed attribute for an ora.* resource. You cannot edit this attribute.

INSTANCE_COUNT

The INSTANCE_COUNT attribute is an internally managed attribute that contains the number of instances that the resource currently has.

INTERNAL_STATE

An internally managed, read-only attribute that describes what, if any, action the policy engine is currently executing on the resource. Possible values and their meanings are as follows:

- STARTING: The policy engine is currently starting the resource
- STOPPING: The policy engine is currently stopping the resource
- CLEANING: The policy engine is currently cleaning the resource
- STABLE: The policy engine is not currently executing any action on the resource

Note, however, that the resource can still be locked as part of some other command.

LAST_SERVER

For `cluster_resource`-type resources, this is an internally managed, read-only attribute that contains the name of the server on which the last `start` action for the resource succeeded.

For `local_resource`-type resources, this is the name of the server to which the resource instance is pinned.

LAST_STATE_CHANGE

An internally managed, read-only attribute that describes when the policy engine registers the current state of the resource. Note that this may either be the timestamp of when state of the resource changed or when the policy engine discovered the state, as occurs when CRSD restarts.

PROFILE_CHANGE_EVENT_TEMPLATE

This is an internally-managed attribute for an `ora.*` resource. You cannot edit this attribute.

RESTART_COUNT

An internally-managed attribute used by the Oracle Clusterware daemon to count the number of attempts to restart a resource, starting from zero up to the value specified in the `RESTART_ATTEMPTS` attribute. You cannot edit this attribute.

STATE

An internally-managed attribute that reflects the current state of the resource as reported by Oracle Clusterware. The state of a resource can be one of the following:

- `ONLINE`: The resource is online and resource monitoring is enabled (see `CHECK_INTERVAL`).
- `OFFLINE`: The resource is offline and only offline resource monitoring is enabled, if configured (see `OFFLINE_CHECK_INTERVAL`).
- `INTERMEDIATE`: The resource is either partially online or was known to be online before and subsequent attempts to determine its state have failed; resource monitoring is enabled (see `CHECK_INTERVAL`).
- `UNKNOWN`: The resource is unmanageable and its current state is unknown; manual intervention is required to resume its operation. A resource in this state is not monitored.

STATE_CHANGE_EVENT_TEMPLATE

This is an internally-managed attribute for an `ora.*` resource. You cannot edit this attribute.

STATE_DETAILS

An internally managed, read-only attribute that contains details about the state of a resource.

The four resource states—`ONLINE`, `OFFLINE`, `UNKNOWN`, and `INTERMEDIATE`—may map to different resource-specific values, such as mounted, unmounted, and open. Resource agent developers can use the STATE_DETAILS attribute to provide a more detailed description of this mapping, resource to the resource state.

Providing details is optional. If details are not provided, then Oracle Clusterware uses only the four possible resource states. Additionally, if the agent cannot provide these details (as may also happen to the value of the resource state), then Oracle Clusterware sets the value of this attribute to provide minimal details about why the resource is in its current state.

TARGET

An internal, read-only attribute that describes the desired state of a resource. Using the `crsctl start` *`resource_name`* or `crsctl stop` *`resource_name`* commands, however, can affect the value of this attribute.

TARGET_SERVER

This is an internally-managed attribute that contains the name of the server where the resource is starting. This value is relevant when the resource is starting.

TYPE

The type of resource indicated when you create a resource. This attribute is required when creating a resource and cannot be changed after the resource is created.

See Also: "Resource Types" on page 9-2 for details of resource types

Deprecated Resource Attributes

The following resource attributes are deprecated in Oracle Clusterware 12*c*:

DEGREE

The number of instances of a cluster resource that can run on a single server.

Examples of Action Scripts for Third-party Applications

This section includes examples of third-party applications using script agents.

Example B–1 shows an action script that fails over the Apache Web server.

Example B–1 Apache Action Script

```
#!/bin/sh

HTTPDCONFLOCATION=/etc/httpd/conf/httpd.conf
WEBPAGECHECK=http://<MyVIP>:80/icons/apache_pb.gif

case $1 in
'start')
    /usr/sbin/apachectl -k start -f $HTTPDCONFLOCATION
   RET=$?
    ;;
sleep(10)
    ;;
'stop')
    /usr/sbin/apachectl -k stop
   RET=$?
    ;;
'clean')
```

```
    /usr/sbin/apachectl -k stop
   RET=$?
    ;;
'check')
    /usr/bin/wget -q --delete-after $WEBPAGECHECK
   RET=$?
    ;;
*)
   RET=0
    ;;
esac
# 0: success; 1 : error
if [ $RET -eq 0 ]; then
exit 0
else
exit 1
fi
```

Example B–2 shows the **xclock** script, which is a simple action script using `xclock` available as a default binary on all Linux and UNIX platforms.

Example B–2   xclock Action Script

```
#!/bin/bash
# start/stop/check script for xclock example
# To test this change BIN_DIR to the directory where xclock is based
# and set the DISPLAY variable to a server within your network.

BIN_DIR=/usr/X11R6/bin
LOG_DIR=/tmp
BIN_NAME=xclock
DISPLAY=yourhost.domain.com:0.0
export DISPLAY
exit_code=0

if [ ! -d $BIN_DIR ]
then
        echo "start failed"
        exit 2
fi

PID1=`ps -ef | grep $BIN_NAME | grep -v grep | grep -v xclock_app | awk '{ print
$2 }'`
case $1 in
'start')
        if [ "$PID1" != "" ]
        then
           status_p1="running"
        else
           if [ -x $BIN_DIR/$BIN_NAME ]
           then
             umask 002
             ${BIN_DIR}/${BIN_NAME} & 2>${LOG_DIR}/${BIN_NAME}.log
             status_p1="started"
           else
             echo `basename $0`": $BIN_NAME: Executable not found"
             exit_code=1
           fi
        fi
```

```
         echo "$BIN_NAME: $status_p1"
         exit $exit_code
        ;;

'stop')
        if [ "${PID1}" != "" ]
        then
           kill -9 ${PID1} && echo "$BIN_NAME daemon killed"
        else
           echo "$BIN_NAME: no running Process!"
        fi
        exit $exit_code
        ;;
'check')
        if [ "$PID1" != "" ]
        then
           echo "running"
           exit 0
        else
           echo "not running"
           exit 1
        fi
        ;;*)
        echo "Usage: "`basename $0`" {start|stop|check}"
        ;;
esac
```

Example B–3 shows an example of a shell script for an agent to monitor a file. When the agent is started, it creates the file (which is specified through an attribute) and when it is stopped, it deletes the file. The CHECK action consists of only checking whether the file exists. The variables with the _CRS_ prefix are attribute values that are provided to the script in its environment.

Example B–3 Action Script Example

```
#!/bin/sh
TOUCH=/bin/touch
RM=/bin/rm
PATH_NAME=/tmp/$_CRS_NAME

#
# These messages go into the CRSD agent log file.
echo " *******   `date` ********** "
echo "Action script '$_CRS_ACTION_SCRIPT' for resource[$_CRS_NAME] called for
action $1"
#

case "$1" in
  'start')
     echo "START entry point has been called.."
     echo "Creating the file: $PATH_NAME"
     $TOUCH $PATH_NAME
     exit 0
     ;;

  'stop')
     echo "STOP entry point has been called.."
     echo "Deleting the file: $PATH_NAME"
     $RM $PATH_NAME
     exit 0
     ;;

  'check')
    echo "CHECK entry point has been called.."
    if [ -e $PATH_NAME ]; then
        echo "Check -- SUCCESS"
        exit 0
    else
        echo "Check -- FAILED"
        exit 1
    fi
    ;;

  'clean')
     echo "CLEAN entry point has been called.."
     echo "Deleting the file: $PATH_NAME"
     $RM -f $PATH_NAME
     exit 0
     ;;

esac
```

C

OLSNODES Command Reference

This appendix describes the syntax and command options for the `olsnodes` command.

This appendix contains the following topics:

Using OLSNODES

This section contains topics which relate to using the OLSNODES command.

Overview

The `olsnodes` command provides the list of nodes and other information for all nodes participating in the cluster.

You can use this command to quickly check that your cluster is operational, and all nodes are registered as members of the cluster. This command also provides an easy method for obtaining the node numbers.

Operational Notes

Usage Information

This command is used by the Cluster Verification Utility (CLUVFY) to obtain a list of node names when the `-n all` option is used.

This command utility is located in the `$ORA_CRS_HOME/bin` directory. You can only use this command if the CRS daemon is started.

Privileges and Security

You can run this command as either the `root` user, the user that installed Oracle Clusterware, or the user that installed Oracle Database.

Summary of the OLSNODES Command

The olsnodes command does not use keywords, but accepts one or more options. The available options are described in Table C–1.

Syntax

```
olsnodes [[-n] [-i] [-s] [-t] [node_name | -l [-p]] | [-c]] [-a] [-g] [-v]
```

If you issue the olsnodes command without any command parameters, the command returns a listing of the nodes in the cluster:

```
[root@node1]# olsnodes
node1
node2
node3
node4
```

Table C–1 OLSNODES Command Options

Command	Description
-n	Lists all nodes participating in the cluster and includes the assigned node numbers.
-i	Lists all nodes participating in the cluster and includes the Virtual Internet Protocol (VIP) address (or VIP address with the node name) assigned to each node.
-s	Displays the status of the node: active or inactive.
-t	Displays node type: pinned or unpinned.
node_name	Displays information for a particular node.
-l [-p]	Lists the local node and includes the private interconnect for the local node. The -p option is only valid when you specify along with the -l option.
-c	Displays the name of the cluster.
-a	Displays only active nodes in the cluster with no duplicates.
-g	Logs cluster verification information with more details.
-v	Logs cluster verification information in verbose mode. Use in debug mode and only at the direction of My Oracle Support.

Examples

Example 1: List the VIP addresses for all nodes currently in the cluster

To list the VIP addresses for each node that is currently a member of the cluster, use the command:

```
[root@node1]# olsnodes -i
node1    168.92.1.1
node2   168.192.2.1
node3   168.192.3.1
node4   168.192.4.1
```

Example 2: List the node names and node numbers for cluster members

To list the node name and the node number for each node in the cluster, use the command:

```
[root@node1]# olsnodes -n
node1    1
node2    2
node3    3
node4    4
```

Example 3: Display node roles for cluster members

To list the node roles for each node in the cluster, use the command:

```
[root@node1]# olsnodes -a
node1    Hub
node2    Hub
node3    Leaf
node4    Leaf
```

D

Oracle Interface Configuration Tool (OIFCFG) Command Reference

The Oracle Interface Configuration Tool (OIFCFG) command-line interface helps you to define and administer network interfaces. You can use OIFCFG commands Oracle Clusterware environments to:

- Allocate and deallocate network interfaces to components
- Direct components to use specific network interfaces
- Retrieve component configuration information

This appendix includes the following topics:

- Starting the OIFCFG Command-Line Interface
- Summary of the OIFCFG Usage

Starting the OIFCFG Command-Line Interface

Before you invoke OIFCFG, ensure that you have started Oracle Clusterware on at least the local node and preferably on all nodes if you intend to include the `-global` option in the command syntax.

> **Note:** To change the global network interface, Oracle Clusterware must be running on all cluster nodes.

Run OIFCFG from the *`Grid_home`*`/bin/` directory as the user who installed the Oracle Clusterware software. For example:

```
$ ./oifcfg
```

Run the `oifcfg -help` command to display online help for OIFCFG.

```
$ ./oifcfg -help
```

Summary of the OIFCFG Usage

This section contains the following topics:

- OIFCFG Command Format
- OIFCFG Commands
- OIFCFG Command Parameters

- OIFCFG Usage Notes
- OIFCFG Examples

OIFCFG Command Format

```
oifcfg iflist [-p] [-n]
oifcfg setif {-node nodename | -global} {if_name/subnet:if_type[,if_type]}[,...]
oifcfg getif [-node nodename | -global] [ -if if_name[/subnet] [-type if_type]]
oifcfg delif {{-node nodename | -global} [if_name[/subnet]] [-force] | -force}
oifcfg [-help]
```

OIFCFG Commands

You can enter any of the OIFCFG commands listed in Table D–1.

Table D–1 OIFCFG Commands

Command	Description

`oifcfg iflist [-p [-n]]`	Shows the available interfaces that you can configure with `setif`. The `iflist` command queries the operating system to find which network interfaces are present on this node. You can specify two options with this command:
	■ `-p`: Displays a heuristic assumption of the interface type (PRIVATE, PUBLIC, or UNKNOWN)
	■ `-n`: Displays the netmask
	If the interface has an IPv6 network, then OIFCFG prints the IPv6 subnet on a separate line, similar to the following:
	`eth1 fec0::80 PUBLIC 64`
	`eth1 10.229.184.0 PUBLIC 255.255.248.0`

`oifcfg setif`	Sets an interface type (public, cluster interconnect, or Oracle ASM) for an interface.
`oifcfg getif`	Displays the interfaces for which an interface type has been defined with the `setif` command, along with the type for that interface.
`oifcfg delif`	Deletes the stored network configuration for global or node-specific interfaces. You can use the `-force` option without specifying the `-node` or `-global` options to delete the stored network configurations from all nodes in the cluster.

OIFCFG Command Parameters

This section lists the parameters for the OIFCFG commands. Note that some parameters are optional, depending on which command you run.

-node *nodename*

The name of the Oracle Clusterware node as listed in the output from the `olsnodes` command. Appendix C, "OLSNODES Command Reference" describes the `olsnodes` command.

-global

A network interface can be stored as a *global interface* (as reported by the `iflist` command) or as a *node-specific interface*:

- An interface is stored as a global interface when all of the nodes of an Oracle Real Application Clusters (Oracle RAC) cluster have the same interface connected to the same subnet. The global interface (and configuring all nodes with the same network interface for each public subnet and the same network interface for each private subnet) is not only the recommended configuration, but it is also the default installation configuration.
- An interface can be stored as a node-specific (local) interface.

> **Note:** Oracle supports interface name wildcards for different interface names across nodes. Oracle *does not* support using `-node` to configure different specific interface names on different nodes.

-if *if_name*
The name by which the interface is configured in the system.

subnet
The subnet address of the interface.

-type *if_type*
One or more comma-delimited interface types: `public`, `cluster_interconnect`, or `asm`.

-help
Display online help for OIFCFG commands.

OIFCFG Usage Notes

- A network interface specification takes the following form:

  `if_name``/`*`subnet`*`:`*`if_type`*

  The specification uniquely identifies the network interface using the:

  – Interface name

    When using the `oifcfg setif` command, you can specify interface names that contain wildcard characters, such as an asterisk (*), which matches any string. However, you must surround interface names that contain wildcards with double quotation marks (`""`).

    > **Note:** Oracle recommends that you *do not* use wildcards in a cluster with pre-Oracle Database 11*g* release 2 (11.2.0.2) databases because Oracle resolves the interface name by expanding the wildcard on the local node and also issues a PRIF-0029 warning every time you run the `oifcfg getif` command.
    >
    > If you use wildcards with Oracle Databases that are pre-11*g* release 2 (11.2.0.2), then those databases must use the `CLUSTER_INTERCONNECTS` parameter, instead.

  – Associated subnet

  – Interface type

    The interface type indicates the purpose for which the network is configured. The supported interface types are:

    - `public`: An interface that can be used for communication with components external to Oracle RAC instances, such as Oracle Net and Virtual Internet Protocol (VIP) addresses.
    - `cluster_interconnect`: A private interface used for the cluster interconnect to provide interinstance or Cache Fusion[1] communication.

    If you set the interface type to `cluster_interconnect`, then it affects instances as they start and changes do not take effect until you restart the instances.

  For example, the following specification identifies `qfe0` as a cluster interconnect located at the address 204.152.65.0:

  ```
  qfe0/204.152.65.0:cluster_interconnect
  ```

- The Oracle Universal Installer uses OIFCFG to identify and display available interfaces.
- The effect of changing the interface names depends on which name you are changing, and whether you are also changing the IP address. In cases where you change only the interface names, the ramifications are minor. If you change the name for the public interface that is stored in the Oracle Cluster Registry (OCR), you must modify the nodeapps for each node. Therefore, you must stop the nodeapps for this change to take effect.
- You must restart Oracle Clusterware on all members of the cluster when you make global changes. For local changes, you need only to perform a node restart. Interconnect changes for the database occur at instance startup. However, the interconnect for Oracle Clusterware might be different.
- Because interconnects are chosen when instances start, just issuing OIFCFG commands does not have an immediate effect on the running system. Instead, changes take effect after restarting the component that might be affected by the command.
- Changes you make to cluster-wide network classifications with OIFCFG require all known cluster nodes to be up so that their Grid Plug and Play profiles can be updated.

OIFCFG Examples

The following examples show some common uses for the OIFCFG commands.

Example 1 Listing the Names of Network Interfaces

You can use OIFCFG to list the interface names and the subnets of all of the interfaces available on the local node by executing the `iflist` keyword, as shown in this example:

```
oifcfg iflist
eth0     172.19.141.0
eth1     172.21.65.0
```

Example 2 Retrieving Network Information

You can also retrieve specific OIFCFG information with a `getif` command.

[1] Cache Fusion is a diskless cache coherency mechanism that provides copies of blocks directly from a holding instance's memory cache to a requesting instance's memory cache.

For example, after you install Oracle Clusterware, you can verify that the public and cluster interconnect have been set to the desired values by entering the following command:

```
$ oifcfg getif
```

This command returns values for global `public` and global `cluster_interconnect`, similar to the following:

```
eth0 172.19.141.0 global public
eth1 172.21.65.0 global cluster_interconnect
```

Example 3 Storing a Global Interface

To store an interface, use the `setif` keyword. For example, to store the interface `eth0`, with the subnet 172.19.141.0, as a global interface (to be used as an interconnect for all of the Oracle RAC instances in your cluster and Oracle Clusterware), use the command:

```
oifcfg setif -global eth0/172.19.141.0:cluster_interconnect
```

> **Note:** Ensure that all nodes are running when you run the `setif` command because Oracle cannot update Grid Plug and Play profiles on nodes that are not running.

Example 4 Deleting the Stored Interface

Use the `oifcfg delif` command to delete the stored configuration for global or node-specific interfaces. A specific node-specific or global interface can be deleted by supplying the interface name, with an optional subnet, on the command line.

> **Note:** If you only enter `oifcfg delif`, with no other arguments given on the command line, then OIFCFG deletes all interfaces on all nodes in the cluster.

For example, the following command deletes the global interface named `eth1` for the subnet 172.21.65.0:

```
oifcfg delif -global eth1/172.21.65.0
```

The following command deletes all of the global interfaces assigned with OIFCFG:

```
oifcfg delif -global
```

E

Oracle Clusterware Control (CRSCTL) Utility Reference

This appendix contains reference information for the Oracle Clusterware Control (CRSCTL) utility.

> **Note:** Do not use CRSCTL commands on Oracle entities (such as resources, resource types, and server pools) that have names beginning with *ora* unless you are directed to do so by My Oracle Support. The Server Control utility (SRVCTL) is the correct utility to use on Oracle entities.

This appendix includes the following topics:

CRSCTL Overview

CRSCTL is an interface between you and Oracle Clusterware, parsing and calling Oracle Clusterware APIs for Oracle Clusterware objects.

CRSCTL provides cluster-aware commands with which you can perform check, start, and stop operations on the cluster. You can run these commands from any node in the cluster on another node in the cluster, or on all nodes in the cluster, depending on the operation.

You can use CRSCTL commands to perform several operations on Oracle Clusterware, such as:

- Starting and stopping Oracle Clusterware resources
- Enabling and disabling Oracle Clusterware daemons
- Checking the health of the cluster
- Managing resources that represent third-party applications

- Integrating Intelligent Platform Management Interface (IPMI) with Oracle Clusterware to provide failure isolation support and to ensure cluster integrity
- Debugging Oracle Clusterware components

Clusterized (Cluster Aware) Commands

You can run clusterized commands on one node to perform operations on another node in the cluster. These are referred to as remote operations. This simplifies administration because, for example, you no longer have to log in to each node to check the status of the Oracle Clusterware on all of your nodes.

Clusterized commands are completely operating system independent; they rely on the OHASD (Oracle High Availability Services daemon). If this daemon is running, then you can perform remote operations, such as the starting, stopping, and checking the status of remote nodes.

Clusterized commands include the following:

- `crsctl check cluster`
- `crsctl start cluster`
- `crsctl stop cluster`

Operational Notes

Usage Information

- The CRSCTL utility is located in the `*Grid_home*/bin` directory. To run CRSCTL commands, type in `crsctl` at the operating system prompt followed by the command and arguments, as shown in the following example:

  ```
  crsctl stop crs
  ```

- There are three categories of CRSCTL commands:
  - Those that you use in either the Oracle Real Application Clusters (Oracle RAC) environment or in the Oracle Restart environment
  - Those that you use in the Oracle RAC environment, only
  - Those that you use in the Oracle Restart environment, only
- Many CRSCTL commands use the `-f` parameter to force the command to run and ignore any checks.

  For example, if you specify the force parameter for the `crsctl stop resource` command on a resource that is running and has dependent resources that are also running, then the force parameter omits the error message and instead stops or relocates all the dependent resources before stopping the resource you reference in the command.
- *Do not* use versions of CRSCTL earlier than 12*c* release 1 (12.1) to manage Oracle Clusterware 12*c*.

Filters

You can use filters to narrow down Oracle Clusterware entities upon which a CRSCTL command operates, as follows:

- Simple filters are attribute-value pairs with an operator.
- Operators must be surrounded by spaces, as shown in the examples.

- You can combine simple filters into expressions called expression filters using Boolean operators.

Supported filter operators are:

`=`
`>`
`<`
`!=`
`co`: Contains
`st`: Starts with
`en`: Ends with

Supported Boolean operators are `AND` and `OR`.

Examples of filters are:

- `TYPE = type1`
- `((TYPE = type1) AND (CHECK_INTERVAL > 50))`
- `(TYPE = type1) AND ((CHECK_INTERVAL > 30) OR (AUTO_START co never))`
- `NAME en network.res`
- `TYPE st ora.db`

Using the eval Command

The `eval` command, when you use it, enables you to simulate a command without making any changes to the system. CRSCTL returns output that informs you what will happen if you run a particular command.

The `eval` commands available are:

- `crsctl eval activate policy`
- `crsctl eval add resource`
- `crsctl eval add server`
- `crsctl eval add serverpool`
- `crsctl eval delete server`
- `crsctl eval delete serverpool`
- `crsctl eval fail resource`
- `crsctl eval modify resource`
- `crsctl eval modify serverpool`
- `crsctl eval relocate resource`
- `crsctl eval relocate server`
- `crsctl eval start resource`
- `crsctl eval stop resource`

> **Note:** CRSCTL can only evaluate third-party resources. Resources with the *.ora* prefix, such as `ora.orcl.db`, must be evaluated using SRVCTL commands.

See Also:

- *Oracle Real Application Clusters Administration and Deployment Guide* for information about using SRVCTL evaluation commands
- "What-If APIs" on page H-12

Using CRSCTL Help

To print the help information for CRSCTL, use the following command:

```
crsctl -help
```

If you want help for a specific command, such as `start`, then enter the command and append `-help` to the end, as shown in the following example:

```
crsctl start -help
```

You can also use the abbreviations `-h` or `-?` (this parameter functions in Linux, UNIX, and Windows environments) instead of `-help`.

Deprecated Subprograms or Commands

Table E–1 lists deprecated commands and their replacements that you can use to perform the same or similar functionality.

Table E–1 Deprecated CRSCTL Commands and Replacements

Deprecated Command	Replacement Commands

`crs_stat`	`crsctl check cluster`
	`crsctl status resource`

`crs_register`	`crsctl add resource`
	`crsctl add type`
	`crsctl modify resource`
	`crsctl modify type`

`crs_unregister`	`crsctl stop resource`
	`crsctl delete resource`

`crs_start`	`crsctl start resource`
	`crsctl start crs`
	`crsctl start cluster`

`crs_stop`	`crsctl stop resource`
	`crsctl stop crs`
	`crsctl stop cluster`

`crs_getperm`	`crsctl getperm resource`
	`crsctl getperm type`

`crs_profile`	`crsctl add resource`
	`crsctl add type`
	`crsctl status resource`
	`crsctl status type`
	`crsctl modify resource`
	`crsctl modify type`

`crs_relocate`	`crsctl relocate resource`

`crs_setperm`	`crsctl setperm resource`
	`crsctl setperm type`

`crsctl add crs administrator`	Use the access control list (ACL) to control who can add server pools.

Table E–1 (Cont.) Deprecated CRSCTL Commands and Replacements

Deprecated Command	Replacement Commands
`crsctl check crsd`	`crsctl check crs`
`crsctl check cssd`	`crsctl check css`
`crsctl check evmd`	`crsctl check evm`
`crsctl debug res log` *`resource_name`*`:`*`level`*	`crsctl set log`

`crsctl set css votedisk`	`crsctl add css votedisk`
	`crsctl delete css votedisk`
	`crsctl query css votedisk`
	`crsctl replace votedisk`

`crsctl start resources`	`crsctl start resource -all`
`crsctl stop resources`	`crsctl stop resource -all`

Dual Environment CRSCTL Commands

You can use the following commands in either the Oracle RAC or the Oracle Restart environments:

- `crsctl add resource`
- `crsctl add type`
- `crsctl add wallet`
- `crsctl check css`
- `crsctl check evm`
- `crsctl delete resource`
- `crsctl delete type`
- `crsctl delete wallet`
- `crsctl eval add resource`
- `crsctl eval fail resource`
- `crsctl eval modify resource`
- `crsctl eval relocate resource`
- `crsctl eval start resource`
- `crsctl eval stop resource`
- `crsctl get hostname`
- `crsctl getperm resource`
- `crsctl getperm type`
- `crsctl modify resource`
- `crsctl modify type`
- `crsctl modify wallet`
- `crsctl query wallet`
- `crsctl relocate resource`

- crsctl restart resource
- crsctl setperm resource
- crsctl setperm type
- crsctl start resource
- crsctl status resource
- crsctl status type
- crsctl stop resource

crsctl add resource

Use the `crsctl add resource` command to register a resource to be managed by Oracle Clusterware. A resource can be an application process, a database, a service, a listener, and so on.

Syntax

```
crsctl add resource resource_name -type resource_type [-file file_path |
   -attr "attribute_name=attribute_value,attribute_name=attribute_value,..."]
  [-i] [-f]
```

Parameters

Table E–2 crsctl add resource Command Parameters

Parameter	Description
resource_name	A short, descriptive name for the resource.
-type *resource_type*	The type of resource that you are adding preceded by the `-type` flag.
-file *file_path*	Path name (either absolute or relative) for a text file containing line-delimited attribute name-value pairs that define the resource.

Table E–2 (Cont.) crsctl add resource Command Parameters

Parameter	Description

`-attr "`*`attribute_name`*`=`
`attribute_value`
	You can specify attributes for a resource you are adding in two different ways:
	■ Following the `-attr` flag, you can specify one or more comma-delimited attribute name-value pairs enclosed in double quotations marks (`""`). For example:
	`-attr "CHECK_INTERVAL=30,START_TIMEOUT=25"`
	Some attributes can have multiple values. In those cases, separate the values with a space and enclose the list of values in single quotation marks. For example:
	`-attr "SERVER_POOL_NAMES=`
	`'ora.pool1 ora.pool2',START_TIMEOUT=25"`
	■ Additionally, you can specify attribute values for resource instances with a particular cardinality value, and with a particular degree value. This method can be useful for applications that are tied to a particular server. Following the `-attr` flag, the syntax is as follows:
	`attribute_name``{@SERVERNAME(`*`server_name`*`)`
	`[@DEGREEID(`*`did`*`)] | @CARDINALITYID(`*`cid`*`)`
	`[@DEGREEID(`*`did`*`)]}=`*`attribute_value`*
	If you specify the `@SERVERNAME(`*`server_name`*`)` syntax, then the attribute value you specify for the attribute you specify is limited to resource instances residing on the server you specify.
	Alternatively, if you specify the `@CARDINALITYID(`*`cid`*`)` syntax, then the attribute value you specify for the attribute you specify is limited to resource instances with a specific cardinality ID (*`cid`*).
	Optionally, you can combine the `@DEGREEID(`*`did`*`)` syntax with either the `SERVERNAME` or `CARDINALITYID` syntax, or both, to limit the attribute value to resources with the specific `DEGREE`.
	Examples:
	`CHECK_INTERVAL@SERVERNAME(node1)=45`
	`STOP_TIMEOUT@CARDINALITYID(2)=65`
	`STOP_TIMEOUT@SERVERNAME(node1)@DEGREEID(2)=65`
	`STOP_TIMEOUT@CARDINALITYID(3)@DEGREEID(2)=65`

`-i`	If you specify `-i`, then the command returns an error if processing this command requires waiting for Oracle Clusterware to unlock the resource or its dependents. Sometimes, Oracle Clusterware locks resources or other objects to prevent commands from interfering with each other.

Table E–2 (Cont.) crsctl add resource Command Parameters

Parameter	Description
`-f`	Use the force parameter: ■ To add a resource that has dependencies on other resources that do not yet exist. The force parameter overrides checks that would prevent a command from being completed. ■ To add a resource if the resource has hard dependencies on other resources and the owner of the resources does not execute permissions on one or more of the dependencies. If you do not specify the force parameter in this case, an error displays. ■ To add resources of `application` type because you may need to move servers into the Generic server pool. If the servers currently host resources that must be stopped, then the force parameter is required

> **See Also:** Appendix B, "Oracle Clusterware Resource Reference" for more information about resources and resource attributes

Usage Notes

- Both the *`resource_name`* and `-type` *`resource_type`* parameters are required. You can create an associated resource type using the `crsctl add type` command.
- Any user can create a resource but only clusterware administrators can create resources of type `local_resource` or resources of type `cluster_resource` that have `SERVER_POOLS=*`.

  Once a resource is defined, its ACL controls who can perform particular operations with it. The Oracle Clusterware administrator list is no longer relevant.

  On Windows, a member of the Administrators group has full control over everything.

  > **See Also:** "`crsctl setperm resource`" on page E-27 for more information about setting ACLs

- If an attribute value for an attribute name-value pair contains commas, then the value must be enclosed in single quotation marks (`''`).
- Following is an example of an attribute file:

```
PLACEMENT=favored
HOSTING_MEMBERS=node1 node2 node3
RESTART_ATTEMPTS@CARDINALITYID(1)=0
RESTART_ATTEMPTS@CARDINALITYID(2)=0
FAILURE_THRESHOLD@CARDINALITYID(1)=2
FAILURE_THRESHOLD@CARDINALITYID(2)=4
FAILURE_INTERVAL@CARDINALITYID(1)=300
FAILURE_INTERVAL@CARDINALITYID(2)=500
CHECK_INTERVAL=2
CARDINALITY=2
```

- Do not use this command for any resources with names that begin with *ora* because these resources are Oracle resources.

Examples

Example 1

To register a VIP as a resource with Oracle Clusterware:

```
$ crsctl add resource app.appvip -type app.appvip.type -attr "RESTART_ATTEMPTS=2,
START_TIMEOUT=100,STOP_TIMEOUT=100,CHECK_INTERVAL=10,
USR_ORA_VIP=172.16.0.0,
START_DEPENDENCIES=hard(ora.net1.network)pullup(ora.net1.network),
STOP_DEPENDENCIES=hard(ora.net1.network)"
```

Example 2

To register a resource based on the `test_type1` resource type:

```
$ crsctl add resource r1 -type test_type1 -attr "PATH_NAME=/tmp/r1.txt"
$ crsctl add resource r1 -type test_type1 -attr "PATH_NAME=/tmp/r2.txt"
```

Example 3

To register a Samba server resource of the `generic_application` resource type, using the `EXECUTABLE_NAMES` attribute:

```
# crsctl add resource my_samba -type generic_application -attr
"EXECUTABLE_NAMES=smbd,START_PROGRAM='/etc/rc.d/init.d/smb start',
STOP_PROGRAM='/etc/rc.d/init.d/smb stop'"
```

Example 4

To register a DNS server of the `generic_application` resource type, using the `EXECUTABLE_NAMES` attribute:

```
# crsctl add resource my_dns -type generic_application -attr
"EXECUTABLE_NAMES=named,START_PROGRAM='/etc/rc.d/init.d/named start',
STOP_PROGRAM='/etc/rc.d/init.d/named stop'"
```

Example 5

To register an Apache web server of the `generic_application` resource type using the PID_FILES attribute:

```
# crsctl add resource my_apache -type generic_application -attr
"START_PROGRAM='/usr/sbin/httpd -k start',STOP_PROGRAM='/usr/sbin/httpd -k stop',
PID_FILES=/etc/httpd/run/httpd.pid"
```

Example 6

To register an application of `generic_application` resource type using environment variables:

```
# crsctl add resource my_app -type generic_application -attr
"START_PROGRAM='/opt/my_app start', EXECUTABLE_NAMES=my_app,
ENVIRONMENT_VARS='USE_NETAPP=no,USE_BACKUP=yes,CLEAN_ON_KILL=yes'"
```

crsctl add type

Use the `crsctl add type` command to create a resource type in Oracle Clusterware.

Syntax

```
crsctl add type type_name -basetype base_type_name {-attr
"ATTRIBUTE=attribute_name | -file file_path,TYPE={string | int}
 [,DEFAULT_VALUE=default_value][,FLAGS=typeFlags"} [-i]
```

Parameters

Table E–3 crsctl add type Command Parameters

Parameter	Description
type_name	A name for the resource type in the form of *xxx.yyy*.type. Resource type names must be unique and cannot be changed after the resource type is registered.
-basetype *base_type_name*	The name of an existing base type. Any resource type that you create must either have `local_resource` or `cluster_resource` as its base resource type.
-attr	You can specify the resource type attributes using the `-attr` argument. Each type attribute definition can contain up to four type attribute keywords that must be displayed in the order shown. Enter a comma-delimited description of one or more resource type attributes enclosed in double quotation marks (`""`). The keywords for an attribute include: **1.** `ATTRIBUTE`: Specify a name for the attribute. The name is case-sensitive and cannot contain spaces. **2.** `TYPE`: Specify whether the attribute type is `integer` or `string`. **3.** `DEFAULT_VALUE`: (Optional) If the attribute is required, then a default value is not required. For attributes that are not required, you must specify a default value that Oracle Clusterware uses when you create resources based on this resource type. **4.** `FLAGS`: (Optional) Specify one or more of the following types, separated by a vertical bar (|): `CONFIG`: After you register a resource with this resource type, you can configure the attribute. `READONLY`: After you register a resource with this resource type, you cannot modify this attribute. `REQUIRED`: You must specify the name and value of this attribute when you create a resource that is based on this resource type. If you specify that this attribute is not required, then Oracle Clusterware uses the default value of this attribute that you specify. `HOTMOD`: If you change the value of an attribute for resources of this type, then the changes are applied immediately with the need to restart the resource. You *cannot* use multiple `-attr` arguments to define multiple arguments for the resource type. Instead, you can specify multiple types within the double quotation marks after the `-attr` flag. For example: `"ATTRIBUTE=FOO,TYPE=integer,DEFAULT_VALUE=0,FLAGS=REQUIRED|HOTMOD,ATTRIBUTE=BAR,TYPE=string"` The preceding example defines two type attributes, `FOO` and `BAR`. When you specify the `ATTRIBUTE` keyword, it ends the previous type attribute (if any) and begins a new type attribute.

Table E–3 (Cont.) crsctl add type Command Parameters

Parameter	Description

`-file` *`file_path`*	Path name (either absolute or relative) for a text file containing line-delimited resource type keyword-value pairs that define the resource type. An example of the contents of the file is:
	`ATTRIBUTE=FOO`
	`TYPE=integer`
	`DEFAULT_VALUE=0`
	`FLAGS=REQUIRED`
	`ATTRIBUTE=BAR`
	`TYPE=string`
	Note: The keywords must be in the following order: `ATTRIBUTE`, `TYPE`, `DEFAULT_VALUE`, and `FLAGS`. When you specify the `ATTRIBUTE` keyword, it ends the previous type attribute (if any) and begins a new type attribute.
	The preceding example defines two type attributes, `FOO` and `BAR`.
	Note: All operators must be surrounded by spaces.
	See Also: "Filters" on page E-2 for more information about operators

`-i`	If you specify `-i`, then the command fails if Oracle Clusterware cannot process the request immediately.

See Also: "Resource Types" on page 9-2 for more information about resource types

Usage Notes

- Both the *`type_name`* and *`base_type_name`* parameters are required
- You can either specify a file containing the type information or you can specify the type information on the command line
- Do not use this command for any resource types with names that begin with *ora* because these resource types are Oracle resource types
- You must have read permissions on the base type

Example

To create a resource type for `demoActionScript`:

```
# crsctl add type test_type1 -basetype cluster_resource
 -attr "ATTRIBUTE=FOO,TYPE=integer,DEFAULT_VALUE=0"
```

crsctl add wallet

Use the `crsctl add wallet` command to create and add users to a wallet.

Syntax

```
crsctl add wallet -type wallet_type [-name name] [-user user_name -passwd]
```

Table E–4 crsctl add wallet

Parameter	Description
`-type wallet_type`	Type of wallet you want to create, such as APPQOSADMIN, APPQOSUSER, APPQOSDB, OSUSER, or CVUDB. ■ **OSUSER**: This wallet type stores a low-privileged Windows user's user name and password that the agent uses when you create a Windows service on a policy-managed database or in general to update the Windows service's password. ■ **CVUDB**: This wallet type stores a database user name and password that the health check component of CVU uses to connect to the database and perform database checks.
`-name name`	You must specify a name for the wallet to create APPQOSDB and CVUDB wallets.
`-user user_name -passwd`	Specify the user name you want to add to the wallet and provide the password through standard input. The user name is required to create an OSUSER wallet.

Usage Notes

- If you are using a policy-managed database, then you must have a wallet. Otherwise, wallets are optional.

Example

To add a wallet:

```
$ crsctl add wallet -type OSUSER -user lp_oracle_home_user -passwd
```

In the preceding example, *`lp_oracle_home_user`* is a low-privileged Oracle home user who owns the home where the policy-managed database was created.

crsctl check css

Use the `crsctl check css` command to check the status of Cluster Synchronization Services. This command is most often used when Oracle Automatic Storage Management (Oracle ASM) is installed on the local server.

Syntax

```
crsctl check css
```

Example

The `crsctl check css` command returns output similar to the following:

```
CRS-4529: Cluster Synchronization Services is online
```

crsctl check evm

Use the `crsctl check evm` command to check the status of the Event Manager.

Syntax

```
crsctl check evm
```

Example

The `crsctl check evm` command returns output similar to the following:

```
CRS-4533: Event Manager is online
```

crsctl delete resource

Use the `crsctl delete resource` command to remove resources from the Oracle Clusterware configuration.

Syntax

```
crsctl delete resource resource_name [-i] [-f]
```

Parameters

Table E–5 crsctl delete resource Command Parameters

Parameter	Description
resource_name	Specify the name of the resource you want to remove or specify a space-delimited list of multiple resources you want to remove.
-i	If you specify `-i`, then the command returns an error if processing this command requires waiting for Oracle Clusterware to unlock the resource or its dependents. Sometimes, Oracle Clusterware locks resources or other objects to prevent commands from interfering with each other.
-f	Use the force parameter to remove either running resources, or remove this resource even though other resources have a hard dependency on it.

Usage Notes

- The *resource_name* parameter is required
- You must have read and write permissions to delete the specified resources
- Do not use this command for any resources with names that begin with *ora* because these resources are Oracle resources

Example

To delete a resource from Oracle Clusterware:

```
# crsctl delete resource myResource
```

crsctl delete type

Use the `crsctl delete type` command to remove resource types from the Oracle Clusterware configuration.

Syntax

```
crsctl delete type type_name [-i]
```

Usage Notes

- The *type_name* parameter is required. You can specify more than one type by separating each type by a space.

- If you specify `-i`, then the command fails if Oracle Clusterware cannot process the request immediately.
- Do not use this command for any resource types with names that begin with *ora* because these resource types are Oracle resource types.

Example

To delete two resource types, run the following command as a user who has write permissions on the resource type:

```
$ crsctl delete type test_type1 test_type2
```

crsctl delete wallet

Use the `crsctl delete wallet` command to remove wallets or users from a wallet.

Syntax

```
crsctl delete wallet -type wallet_type [-name name] [-user user_name]
```

Table E–6 crsctl delete wallet

Parameter	Description
`-type` *`wallet_type`*	Type of wallet you want to remove, such as APPQOSADMIN, APPQOSUSER, APPQOSDB, OSUSER, or CVUDB. ■ **OSUSER**: This wallet type stores a low-privileged Windows user's user name and password that the agent uses when you create a Windows service on a policy-managed database or in general to update the Windows service's password. ■ **CVUDB**: This wallet type stores a database user name and password that the health check component of CVU uses to connect to the database and perform database checks.
`-name` *`name`*	You must specify the name of the wallet to remove an APPQOSDB wallet.
`-user` *`user_name`*	You must specify a user name to remove a user from an OSUSER wallet.

Example

To delete a user from the OSUSER wallet:

```
$ crsctl delete wallet -type OSUSER -user lp_oracle_home_user
```

In the preceding example, *`lp_oracle_home_user`* is a low-privileged Oracle home user who owns the home where the policy-managed database was created. Additionally, the command does not delete the wallet if it contains other users.

crsctl eval add resource

Use the `crsctl eval add resource` command to predict the effects of adding a resource without making changes to the system. This command may be useful to application administrators.

Syntax

```
crsctl eval add resource resource_name -type resource_type
```

```
    [-attr "attribute_name=attribute_value[,attribute_name=attribute_value[,..."]]
    | -file file_path] [-f]
```

Parameters

> **See Also:** "crsctl add resource" on page E-6 for descriptions of the -type, -attr, and -file parameters

Table E–7 crsctl eval add resource Command Parameters

Parameter	Description
`-f`	Specify this parameter to evaluate what happens if you run the command with the force parameter.

crsctl eval fail resource

Use the `crsctl eval fail resource` command to predict the consequences of a resource failing.

Syntax

```
crsctl eval fail resource {resource_name | -w "filter"} [-n server]
```

Parameters

Table E–8 crsctl eval fail resource Command Parameters

Parameter	Description
resource_name	The name of a resource for which you want to simulate a failure.
`-w "`*filter*`"`	Specify a resource filter that Oracle Clusterware uses to limit the number of resources evaluated. The filter must be enclosed in double quotation marks (`""`). Examples of resource filters include: ■ `"TYPE == cluster_resource"`: This filter limits Oracle Clusterware to relocate only resources of `cluster_resource` type ■ `"CHECK_INTERVAL > 10"`: This filter limits Oracle Clusterware to relocate resources that have a value greater than 10 for the `CHECK_INTERVAL` resource attribute ■ `"(CHECK_INTERVAL > 10) AND (NAME co 2)"`: This filter limits Oracle Clusterware to relocate resources that have a value greater than 10 for the `CHECK_INTERVAL` resource attribute *and* the name of the resource contains the number `2` **Note:** All operators must be surrounded by spaces. **See Also:** "Filters" on page E-2 for more information about operators
`-n` *server*	Specify the name of the server on which the resource that you want to simulate a failure resides.

Example

This command returns output similar to the following:

```
$ crsctl eval fail res cs1
```

```
Stage Group 1:
--------------------------------------------------------------------------------
Stage Number    Required       Action
--------------------------------------------------------------------------------

     1              Y   Resource 'cs1' (1/1) will be in state
                        [ONLINE|INTERMEDIATE] on server
                        [mjkeenan_node_0]
                    Y   Resource 'cs1' (2/1) will be in state
                        [ONLINE|INTERMEDIATE] on server
                        [mjkeenan_node_1]

--------------------------------------------------------------------------------
```

crsctl eval modify resource

Use the `crsctl eval modify resource` command to predict the effects of modifying a resource without making changes to the system.

Syntax

```
crsctl eval modify resource resource_name -attr "attribute_name=attribute_value"
    [-f]
```

Parameters

Table E–9 crsctl eval modify resource Command Parameters

Parameter	Description
resource_name	The name of the resource you want to modify.

Table E–9 (Cont.) crsctl eval modify resource Command Parameters

Parameter	Description

`-attr "`*`attribute_name`*`=`
`attribute_value``"`
	You can specify attributes for a resource you want to modify in two different ways:
	■ Following the `-attr` flag, you can specify one or more comma-delimited attribute name-value pairs to modify enclosed in double quotation marks (`""`). For example:
	`-attr "CHECK_INTERVAL=30, START_TIMEOUT=25"`
	Some attributes can have multiple values. In those cases, separate the values with a space and enclose the list of values in single quotation marks. For example:
	`-attr "SERVER_POOL_NAMES=`
	`'ora.pool1 ora.pool2',START_TIMEOUT=25"`
	■ Alternatively, you can specify attribute values for resources on a particular server, with a particular cardinality value, and with a particular degree value. This method can be useful for applications that are somehow tied to a particular server. Following the `-attr` flag, the syntax is as follows:
	`attribute_name``{@SERVERNAME(`*`server_name`*`)`
	`[@DEGREEID(`*`did`*`)] | @CARDINALITYID(`*`cid`*`)`
	`[@DEGREEID(`*`did`*`)]}=`*`attribute_value`*
	If you specify the `@SERVERNAME(`*`server_name`*`)` syntax, then the attribute value you specify for the attribute you specify is limited to resources residing on the server you specify.
	Alternatively, if you specify the `@CARDINALITYID(`*`cid`*`)` syntax, then the attribute value you specify for the attribute you specify is limited to resource instances with a specific cardinality ID (*`cid`*).
	Optionally, you can combine the `@DEGREEID(`*`did`*`)` syntax with either the `SERVERNAME` or `CARDINALITYID` syntax, or both, to limit the attribute value to resources with the specific `DEGREE`.
	Examples:
	`CHECK_INTERVAL@SERVERNAME(node1)=45`
	`STOP_TIMEOUT@CARDINALITYID(2)=65`
	`STOP_TIMEOUT@SERVERNAME(node1)@DEGREEID(2)=65`
	`STOP_TIMEOUT@CARDINALITYID(3)@DEGREEID(2)=65`

`-f`	Specify this parameter to evaluate what happens if you run the command with the force parameter.

See Also: Appendix B, "Oracle Clusterware Resource Reference" for more information about resources and resource attributes

crsctl eval relocate resource

Use the `crsctl eval relocate resource` command to simulate relocating a resource without making changes to the system.

Syntax

```
crsctl eval relocate resource {resource_name | -all} {-s source_server |
-w "filter"} [-n destination_server] [-f]
```

Parameters

> **See Also:** "crsctl relocate resource" on page E-24 for descriptions of the parameters used with this command

crsctl eval start resource

Use the `crsctl eval start resource` command to predict the effects of starting a resource without making changes to the system.

Syntax

```
crsctl eval start resource {resource_name [...] | -w "filter" | -all}
   [-n server_name] [-f]
```

Parameters

> **See Also:** "crsctl start resource" on page E-29 for descriptions of the parameters used with this command

crsctl eval stop resource

Use the `crsctl eval stop resource` command to predict the effects of stopping a resource without making changes to the system.

Syntax

```
crsctl eval stop resource {resource_name [...] | -w "filter" | -all} [-f]
```

Parameters

> **See Also:** "crsctl stop resource" on page E-34 for descriptions of the parameters used with this command

crsctl get hostname

Use the `crsctl get hostname` command to retrieve the host name of the local server.

Syntax

```
crsctl get hostname
```

Example

Oracle Clusterware returns the host name of the local server:

```
$ crsctl get hostname
node2
```

crsctl getperm resource

Use the `crsctl getperm resource` command to display the user and group permissions for the specified resource.

Syntax

```
crsctl getperm resource resource_name [ {-u user_name | -g group_name} ]
```

See Also: Appendix B, "Oracle Clusterware Resource Reference" for more information about resources and resource attributes

Parameters

Table E–10 crsctl getperm resource Command Parameters

Parameter	Description
`resource_name`	Specify the name of the resource for which you want to obtain permissions.
`-u user_name`	If you specify `-u`, then Oracle Clusterware obtains permissions for a particular user.
`-g group_name`	If you specify `-g`, then Oracle Clusterware obtains permissions for a particular group.

Usage Notes

- The `resource_name` parameter is required
- You must have read permission on the specified resources to obtain their permissions
- Do not use this command for any resources with names that begin with *ora* because these resources are Oracle resources

Example

The `crsctl getperm resource` command returns output similar to the following, depending on the command option you choose:

```
$ crsctl getperm resource app.appvip

Name: app.appvip
owner:root:rwx,pgrp:oinstall:rwx,other::r--

$ crsctl getperm resource app.appvip -u oracle

Name: app.appvip
rwx

$ crsctl getperm resource app.appvip -g dba

Name: app.appvip
r--
```

crsctl getperm type

Use the `crsctl getperm type` command to obtain permissions for a particular resource type.

Syntax

```
crsctl getperm type resource_type  [-u user_name] | [-g group_name]
```

See Also: "Resource Types" on page 9-2 for more information about resource types

Parameters

Table E–11 crsctl getperm type Command Parameters

Parameter	Description
resource_type	Specify the resource type for which you want to obtain permissions.
-u *user_name*	If you specify -u, then Oracle Clusterware obtains permissions for a particular user.
-g *group_name*	If you specify -g, then Oracle Clusterware obtains permissions for a particular group.

Usage Notes

- The *resource_type* parameter is required
- Do not use this command for any resource types with names that begin with *ora* because these resource types are Oracle resource types

Example

The `crsctl getperm type` command returns output similar to the following:

```
$ crsctl getperm type app.appvip.type

Name: app.appvip.type
owner:root:rwx,pgrp:oinstall:rwx,other::r--
```

crsctl modify resource

Use the `crsctl modify resource` command to modify the attributes of a particular resource in Oracle Clusterware.

Syntax

```
crsctl modify resource resource_name -attr "attribute_name=attribute_value"
[-i] [-f] [-delete]
```

Parameters

Table E–12 crsctl modify resource Command Parameters

Parameter	Description
resource_name	The name of the resource you want to modify.

Table E–12 (Cont.) crsctl modify resource Command Parameters

<table>
<tr><th>Parameter</th><th>Description</th></tr>
<tr><td>-attr "<i>attribute_name</i>=
<i>attribute_value</i>"</td><td>You can specify attributes for a resource you want to modify in two different ways:
■ Following the -attr flag, you can specify one or more comma-delimited attribute name-value pairs to modify enclosed in double quotations marks (""). For example:
-attr "CHECK_INTERVAL=30, START_TIMEOUT=25"
Some attributes can have multiple values. In those cases, separate the values with a space and enclose the list of values in single quotation marks. For example:
-attr "SERVER_POOL_NAMES=
'ora.pool1 ora.pool2',START_TIMEOUT=25"
■ Alternatively, you can specify attribute values for resources on a particular server, with a particular cardinality value, and with a particular degree value. This method can be useful for applications that are somehow tied to a particular server. Following the -attr flag, the syntax is as follows:
<i>attribute_name</i>{@SERVERNAME(<i>server_name</i>)
[@DEGREEID(<i>did</i>)] | @CARDINALITYID(<i>cid</i>)
[@DEGREEID(<i>did</i>)]}=<i>attribute_value</i>
If you specify the @SERVERNAME(<i>server_name</i>) syntax, then the attribute value you specify for the attribute you specify is limited to resources residing on the server you specify.
Alternatively, if you specify the @CARDINALITYID(<i>cid</i>) syntax, then the attribute value you specify for the attribute you specify is limited to resource instances with a specific cardinality ID (<i>cid</i>).
Optionally, you can combine the @DEGREEID(<i>did</i>) syntax with either the SERVERNAME or CARDINALITYID syntax, or both, to limit the attribute value to resources with the specific DEGREE.
Examples:
CHECK_INTERVAL@SERVERNAME(node1)=45
STOP_TIMEOUT@CARDINALITYID(2)=65
STOP_TIMEOUT@SERVERNAME(node1)@DEGREEID(2)=65
STOP_TIMEOUT@CARDINALITYID(3)@DEGREEID(2)=65</td></tr>
<tr><td>-i</td><td>If you specify -i, then the command returns an error if processing this command requires waiting for Oracle Clusterware to unlock the resource or its dependents. Sometimes, Oracle Clusterware locks resources or other objects to prevent commands from interfering with each other.</td></tr>
<tr><td>-f</td><td>Use the -f parameter when:
■ The resource has a hard dependency on a non-existing resource
■ The owner of the resource does not have execute permissions on one or more hard dependencies
■ The modification results in servers being moved into the Generic pool and resources being stopped or relocated to accomplish the server move</td></tr>
<tr><td>-delete</td><td>If you specify the -delete parameter, then Oracle Clusterware deletes the named attribute.</td></tr>
</table>

> **See Also:** Appendix B, "Oracle Clusterware Resource Reference" for more information about resources and resource attributes

Usage Notes

- The *resource_name* parameter is required
- If an attribute value for an attribute name-value pair contains commas, then the value must be enclosed in single quotation marks (''). For example:

  ```
  "START_DEPENDENCIES='hard(res1,res2,res3)'"
  ```

- You must have read and write permissions on the specified resources to modify them
- Do not use this command for any resources with names that begin with *ora* because these resources are Oracle resources

Example

To modify the attributes of the `appsvip` resource:

```
$ crsctl modify resource appsvip -attr USR_ORA_VIP=10.1.220.17 -i
```

crsctl modify type

Use the `crsctl modify type` command to modify an existing resource type.

Syntax

```
crsctl modify type type_name -attr "ATTRIBUTE=attribute_name,TYPE={string | int}
[,DEFAULT_VALUE=default_value [,FLAGS=[READONLY][| REQUIRED]]" [-i] [-f]]
```

Parameters

Table E–13 crsctl modify type Command Parameters

Parameter	Description
type_name	Specify the name of the resource type you want to modify. You cannot modify resource type names.
`-attr`	You can modify the following resource type keywords: ■ `TYPE` ■ `DEFAULT_VALUE` ■ `FLAGS` **Note:** Although you must specify the `ATTRIBUTE` keyword, you *cannot* modify it. **See Also:** Table E–3, " crsctl add type Command Parameters" for descriptions of these keywords
`-i`	If you specify the `-i` parameter, then the command fails if Oracle Clusterware cannot process the request immediately.

> **See Also:** "Resource Types" on page 9-2 for more information about resource types

Usage Notes

- The *type_name* parameter is required

- Do not use this command for any resource types with names that begin with *ora* because these resource types are Oracle resource types

Example

The following example modifies the two type attributes `FOO` and `BAR`:

```
$ crsctl modify type myType.type -attr "ATTRIBUTE=FOO,DEFAULT_VALUE=0
ATTRIBUTE=BAR,DEFAULT_VALUE=baz"
```

crsctl modify wallet

Use the `crsctl modify wallet` command to modify the password for a specific user in a specific wallet.

Syntax

```
crsctl modify wallet -type wallet_type [-name name] [-user user_name -passwd]
```

Table E–14 crsctl modify wallet

Parameter	Description
`-type` *`wallet_type`*	Specify the type of wallet you want to modify, such as APPQOSADMIN, APPQOSUSER, APPQOSDB, OSUSER, or CVUDB. ■ **OSUSER**: This wallet type stores a low-privileged Windows user's user name and password that the agent uses when you create a Windows service on a policy-managed database or in general to update the Windows service's password. ■ **CVUDB**: This wallet type stores a database user name and password that the health check component of CVU uses to connect to the database and perform database checks.
`-name` *`name`*	You must specify the wallet name to modify an APPQOSDB wallet.
`-user` *`user_name`* `-passwd`	You must specify the user name for whom you want to modify the password. Modify the password through standard input.

Usage Notes

- You cannot use this command to change a user name.

Example

To modify the password of a low-privileged Oracle home user:

```
$ crsctl modify wallet -type OSUSER -user lp_oracle_home_user -passwd
```

crsctl query wallet

Use the `crsctl query wallet` command to query low-privileged users from a wallet.

Syntax

```
crsctl query wallet -type wallet_type [-name name] [-user user_name] [-all]
```

Table E–15 crsctl query wallet

Parameter	Description
`-type` *`wallet_type`*	Type of wallet you want to query, such as APPQOSADMIN, APPQOSUSER, APPQOSDB, OSUSER, or CVUDB. ■ **OSUSER**: This wallet type stores a low-privileged Windows user's user name and password that the agent uses when you create a Windows service on a policy-managed database or in general to update the Windows service's password. ■ **CVUDB**: This wallet type stores a database user name and password that the health check component of CVU uses to connect to the database and perform database checks.
`-name` *`name`*	You must specify the name of the wallet to query an APPQOSDB wallet.
`-user` *`user_name`*	You must specify a user name to query a user from an OSUSER wallet.
`-all`	Specify `-all` to list all of the users in a specific wallet.

Example

To list all of the users in the OSUSER wallet:

```
$ crsctl query wallet -type OSUSER -all
```

crsctl relocate resource

Use the `crsctl relocate resource` command to relocate resources to another server in the cluster.

Syntax

```
crsctl relocate resource {resource_name | resource_name | -all -s source_server |
-w "filter"} [-n destination_server] [-k cid] [-env "env1=val1,env2=val2,..."]
[-i] [-f]
```

Parameters

Table E–16 crsctl relocate resource Command Parameters

Parameter	Description
`resource_name`	The name of a resource you want to relocate.

`resource_name` `| -all`
`-s` *`source_server`*
	Specify one particular or all resources located on a particular server *from* which you want to relocate those resources.

Table E–16 (Cont.) crsctl relocate resource Command Parameters

Parameter	Description
`-w "filter"`	Specify a resource filter that Oracle Clusterware uses to limit the number of resources relocated. The filter must be enclosed in double quotation marks (`""`). Examples of resource filters include: ■ `"TYPE == cluster_resource"`: This filter limits Oracle Clusterware to relocate only resources of `cluster_resource` type ■ `"CHECK_INTERVAL > 10"`: This filter limits Oracle Clusterware to relocate resources that have a value greater than 10 for the `CHECK_INTERVAL` resource attribute ■ `"(CHECK_INTERVAL > 10) AND (NAME co 2)"`: This filter limits Oracle Clusterware to relocate resources that have a value greater than 10 for the `CHECK_INTERVAL` resource attribute *and* the name of the resource contains the number `2` **See Also:** "Filters" on page E-2 for more information
`-n destination_server`	Specify the name of the server *to* which you want relocate resources. If you do not specify a destination server, then Oracle Clusterware relocates the resources to the best server according to the attribute profile of each resource.
`-k cid`	Specify the resource cardinality ID. If you specify this parameter, then Oracle Clusterware relocates the resource instance that have the cardinality you specify.
`-env "env1=val1, env2=val2,..."`	You can optionally override one or more resource profile attribute values for this command. If you specify multiple environment name-value pairs, then you must separate each pair with a comma and enclose the entire list in double quotation marks (`""`).
`-i`	If you specify `-i`, then the command returns an error if processing this command requires waiting for Oracle Clusterware to unlock the resource or its dependents. Sometimes, Oracle Clusterware locks resources or other objects to prevent commands from interfering with each other.
`-f`	Specify the `-f` parameter to force the relocating of the resource when it has other resources running that depend on it. Dependent resources are relocated or stopped when you use this parameter. **Note:** When you are relocating resources that have cardinality greater than 1, you must use either `-k` or `-s` to narrow down which resource instances are to be relocated.

Usage Notes

- Any one of the three following options is required to specify which resources you want to relocate:
  - You can specify one particular resource to relocate.
  - Or you can specify one particular or all the resources to relocate from a particular source server.
  - Thirdly, you can specify a resource filter that Oracle Clusterware uses to match resources to relocate.

- If a resource has a degree ID greater than 1, then Oracle Clusterware relocates all instances of the resource.
- You must have read and execute permissions on the specified resources to relocate them
- Do not use this command for any resources with names that begin with *ora* because these resources are Oracle resources.

Example

To relocate one particular resource from one server to another:

```
# crsctl relocate resource myResource1 -s node1 -n node3
```

crsctl restart resource

Use the `crsctl restart resource` command to restart idle resources in the cluster, instead of having to run two commands to stop and start the resource.

Syntax

```
crsctl restart resource {resource_name [...] | -w "filter"} [-k cid] [-d did]
   [-env "env1=val1,env2=val2,..."] [-i] [-f]
```

Parameters

Table E–17 crsctl restart resource Command Parameters

Parameter	Description
resource_name [...]	One or more space-delimited resource names to restart.
-w *filter*	Specify a resource filter surrounded by double quotation marks ("") that Oracle Clusterware uses to match resources. For example, `-w "TYPE = ora.database.type"` or `-w "NAME = cs1"`. **See Also:** "Filters" on page E-2 for more information
-k *cid*	Specify the resource cardinality ID. If you specify this parameter, then Oracle Clusterware restarts the resource instances that have the cardinality you specify.
-d *did*	Specify the resource degree ID. If you specify this parameter and the degree ID is greater than 1, then Oracle Clusterware restarts all resource instances that meet this criteria. **Note:** You cannot use the `-d` parameter without specifying the `-k` parameter.
-env "*env1*=*val1*, *env2*=*val2*,..."	You can optionally override one or more resource profile attribute values with the `-env` command parameter. If you specify multiple environment name-value pairs, then you must separate each pair with a comma and enclose the entire list in double quotation marks ("").
-i	If you specify `-i`, then the command returns an error if processing this command requires waiting for Oracle Clusterware to unlock the resource or its dependents. Sometimes, Oracle Clusterware locks resources or other objects to prevent commands from interfering with each other.

Table E–17 (Cont.) crsctl restart resource Command Parameters

Parameter	Description
`-f`	Use the `-f` parameter to relocate a resource running on another server on which the resource you want to restart has a hard start dependency. If you do not specify the force parameter in this case, then the start command fails.

Usage Notes

- Any one of the three following options is required to specify which resources you want to restart:
  - You can specify one or more resources to restart
  - You can specify a resource filter that Oracle Clusterware uses to match resources to restart
- You must have read and execute permissions on the specified resources to restart them
- Do not use this command to restart any resources with names that begin with *ora* because these resources are Oracle resources

Example

To restart a resource:

```
# crsctl restart resource myResource -s pool1 pool2
```

crsctl setperm resource

Use the `crsctl setperm resource` command to set permissions for a particular resource.

Syntax

```
crsctl setperm resource resource_name {-u acl_string | -x acl_string |
-o user_name | -g group_name}
```

Parameters

Table E–18 crsctl setperm resource Command Parameters

Parameter	Description
resource_name	Specify the name of the resource for which you want to set permissions.

Table E–18 (Cont.) crsctl setperm resource Command Parameters

Parameter	Description

`{-u | -x | -o | -g}`	You can set only one of the following permissions for a resource:
	- `-u` *`acl_string`*: You can update the access control list (ACL) for a resource
	- `-x` *`acl_string`*: You can delete the ACL for a resource
	- `-o` *`user_name`*: You can change the owner of a resource by entering a user name
	- `-g` *`group_name`*: You can change the primary group of a resource by entering a group name
	Specify a user, group, or other ACL string, as follows:
	`user:`*`user_name`*`[:`*`readPermwritePermexecPerm`*`] |`
	`group:`*`group_name`*`[:`*`readPermwritePermexecPerm`*`] |`
	`other[::`*`readPermwritePermexecPerm`*`]`
	- `user`: User ACL
	- `group`: Group ACL
	- `other`: Other ACL
	- `readPerm`: Read permission for the resource; the letter `r` grants a user, group, or other read permission, the minus sign (-) denies read permission
	- `writePerm`: Write permission for the resource; the letter `w` grants a user, group, or other write permission, the minus sign (-) denies write permission
	- `execPerm`: Execute permission for the resource; the letter `x` grants a user, group, or other execute permission, the minus sign (-) denies execute permission

See Also: Appendix B, "Oracle Clusterware Resource Reference" for more information about resources and resource attributes

Usage Notes

- Do not use this command for any resources with names that begin with *ora* because these resources are Oracle resources.
- You must have read and write permissions on the specified resources to set their permissions

Example

To grant read, write, and execute permissions on a resource for user Scott:

```
$ crsctl setperm resource myResource -u user:scott:rwx
```

crsctl setperm type

Use the `crsctl setperm type` command to set permissions resources of a particular resource type.

Syntax

```
crsctl setperm type resource_type_name {-u acl_string | -x acl_string |
-o user_name | -g group_name}
```

Parameters

Table E–19 crsctl setperm type Command Parameters

Parameter	Description
resource_type_name	Specify the name of the resource type for which you want to set permissions.
{-u | -x | -o | -g}	You can specify only one of the following parameters for a resource type: ■ -u *acl_string*: You can update the access control list (ACL) for a resource type ■ -x *acl_string*: You can delete the ACL for a resource type ■ -o *user_name*: You can change the owner of a resource type by entering a user name ■ -g *group_name*: You can change the primary group of a resource type by entering a group name Specify a user, group, or other ACL string, as follows: user:*user_name*[:*readPermwritePermexecPerm*] | group:*group_name*[:*readPermwritePermexecPerm*] | other[::*readPermwritePermexecPerm*] ■ user: User ACL ■ group: Group ACL ■ other: Other ACL ■ readPerm: Read permission for the resource type; the letter r grants a user, group, or other read permission, the minus sign (-) denies read permission ■ writePerm: Write permission for the resource type; the letter w grants a user, group, or other write permission, the minus sign (-) denies write permission ■ execPerm: Execute permission for the resource type; the letter x grants a user, group, or other execute permission, the minus sign (-) denies execute permission

Usage Notes

- The *resource_type_name* parameter is required
- You must have read and write permissions on the specified resources to set their permissions
- Do not use this command for any resource types with names that begin with *ora* because these resource types are Oracle resource types

Example

To grant read, write, and execute permissions on a resource type for user Scott:

```
$ crsctl setperm type resType -u user:scott:rwx
```

crsctl start resource

Use the `crsctl start resource` command to start many idle resources on a particular server in the cluster.

Syntax

```
crsctl start resource {resource_name [...] | -w "filter" | -all}
   [-n server_name | -s server_pool_names] [-k cid] [-d did]
   [-env "env1=val1,env2=val2,..."] [-begin] [-end] [-i] [-f] [-l]
```

Parameters

Table E–20 crsctl start resource Command Parameters

Parameter	Description
`resource_name [...]`	One or more space-delimited resource names to start.
`-w "filter"`	Specify a resource filter surrounded by double quotation marks (`""`) that Oracle Clusterware uses to match resources. For example, `-w "TYPE = ora.database.type"` or `-w "NAME = cs1"`. **See Also:** "Filters" on page E-2 for more information
`-all`	Use this parameter to start all resources in the cluster.
`-n server_name`	Specify the name of the server on which the resources you want to start reside. If you do not specify a server, then Oracle Clusterware starts the resources on the best server according to the attribute profile of each resource.
`-s server_pool_names`	Specify a single server pool name or a space-delimited list of server pools in which a resource resides that you want to start.
`-k cid`	Specify the resource cardinality ID. If you specify this parameter, then Oracle Clusterware starts the resource instances that have the cardinality you specify.
`-d did`	Specify the resource degree ID. If you specify this parameter and the degree ID is greater than 1, then Oracle Clusterware starts all resource instances that meet this criteria. **Note:** You cannot use the `-d` parameter without specifying the `-k` parameter.
`-env "env1=val1, env2=val2,..."`	You can optionally override one or more resource profile attribute values with the `-env` command parameter. If you specify multiple environment name-value pairs, then you must separate each pair with a comma and enclose the entire list in double quotation marks (`""`).
`-begin`	You can specify this parameter to begin a transparent HA action.
`-end`	You can specify this parameter to end a transparent HA action.
`-i`	If you specify `-i`, then the command returns an error if processing this command requires waiting for Oracle Clusterware to unlock the resource or its dependents. Sometimes, Oracle Clusterware locks resources or other objects to prevent commands from interfering with each other.
`-f`	Use the `-f` parameter to relocate a resource running on another server on which the resource you want to start has a hard start dependency. If you do not specify the force parameter in this case, then the start command fails.
`-l`	Use the `-l` parameter to leave the resources in the state they were in if the start command fails.

Usage Notes

- Any one of the three following options is required to specify which resources you want to start:
  - You can specify one or more resources to start
  - You can specify a resource filter that Oracle Clusterware uses to match resources to start
  - You can specify the `-all` parameter to start all resources on the specified server
- You must have read and execute permissions on the specified resources to start them
- Do not use this command to start any resources with names that begin with *ora* because these resources are Oracle resources
- Oracle does not support starting managed applications outside of the Oracle Grid Infrastructure

Example

To start a resource:

```
# crsctl start resource myResource -n server1
```

crsctl status resource

Use the `crsctl status resource` command to obtain the status and configuration information of many particular resources.

Syntax

Use this command, depending on how you want the information about the status of the resource returned.

To check the status of specific resources:

```
crsctl status resource resource_name [...] | -w "filter" [-p | -v] | [-f | -l |
-g]
  [[-k cid | -n server_name] [ -e [-p | -v]] [-d did]] | [-s -k cid [-d did]]
```

To print the status of the resources in tabular form:

```
crsctl status resource resource_name [...] | -w "filter" -t
```

To print a list of the resource dependencies:

```
crsctl status resource [resource_name [...]] -dependency [-stop | -pullup]
```

Parameters

Table E–21 crsctl status resource Command Parameters

Parameter	Description
resource_name [...] | -w "*filter*"	One or more space-delimited resource names of which you want to check the status. Or you can specify a resource filter that Oracle Clusterware uses to limit the number of resources displayed. The filter must be enclosed in double quotation marks (""). Values that contain parentheses or spaces must be enclosed in single quotation marks (''). Operators must be surrounded by spaces. Examples of resource filters include: ■ "TYPE == cluster_resource": This filter limits the display to only resources of cluster_resource type. ■ "CHECK_INTERVAL > 10": This filter limits the display to resources that have a value greater than 10 for the CHECK_INTERVAL resource attribute ■ "(CHECK_INTERVAL > 10) AND (NAME co 2)": This filter limits the display to resources that have a value greater than 10 for the CHECK_INTERVAL resource attribute *and* the name of the resource contains the number 2. ■ "START_DEPENDENCIES='hard(appsvip)'": This filter limits the display to resources that have a hard start dependency on the appsvip resource. **See Also:** "Filters" on page E-2 for more information
[-p | -v] | [-f | -l | -g]	You can optionally specify the following parameters: ■ Specify either the -p parameter to display the static configuration of the resource or the -v parameter to display the run-time configuration of the resource. ■ Specify the -f parameter to display the full configuration of the resource; or specify the -l parameter to display all cardinal and degree values of the resource; or specify the -g parameter to check whether the specified resources are registered
[[-k *cid* | -n *server_name*] [-e [-p | -v]] [-d *did* | [-s -k cid [-d *did*]]]	You can specify one of the following two options: ■ Specify the -k *cid* parameter to specify a cardinality ID of the resources you want to query. Or you can specify the -n parameter to specify a particular server on which to check resources. Optionally, you can specify the -d parameter with the -n parameter to specify the degree ID of resources you want to check. If you specify a degree ID greater than 1, then Oracle Clusterware checks all resource instances on the server that meet this criteria. Use the -e parameter to evaluate the special values of a resource instance. You must also specify -p or -v with the -e parameter. ■ Specify the -s parameter with the -k parameter to obtain a list of target servers for relocation. You can further limit the output by specifying a degree ID with the -d parameter.
-t	Specify the -t parameter to display the output in tabular form.

Table E–21 (Cont.) crsctl status resource Command Parameters

Parameter	Description
`-dependency [-stop | -pullup]`	Specify the `-dependency` parameter to display resource dependencies. If you do not specify either the `-stop` or `-pullup` option, then CRSCTL displays the start dependencies of the resource. Use either of the following options with the `-dependency` parameter: ■ Specify the `-stop` parameter to display resource stop dependencies. ■ Specify the `-pullup` parameter to display resource pull up dependencies.

Usage Notes

- Either a space-delimited list of resources or a resource filter is required.
- You must have read permissions on the specified resources to obtain their status.
- Use `crsctl status resource` to query the status information of any resource deployed in the cluster. Oracle recommends, however, that you use the respective SRCVTL command to query the status information of Oracle (`ora.*`) resources.

Examples

The `crsctl status resource` command returns output similar to the following:

```
$ crsctl status resource ora.staii14.vip

NAME=ora.staii14.vip
TYPE=ora.cluster_vip_net1.type
TARGET=ONLINE
STATE=ONLINE on staii14
```

The following example shows the start dependencies for a resource named `ora.newdb.db`:

```
$ crsctl status resource ora.newdb.db -dependency ora.newdb.db(ora.database.type)

  ora.ACFS_DG1.dg(ora.diskgroup.type)[hard,pullup]
    ora.asm(ora.asm.type)[hard,pullup]
      ora.LISTENER.lsnr(ora.listener.type)[weak]
        type:ora.cluster_vip_net1.type[hard:type,pullup:type]
          ora.net1.network(ora.network.type)[hard,pullup]
  ora.dbhome_dg.dbhome_dg_v.acfs(ora.acfs.type)[hard,pullup]
    ora.asm(ora.asm.type)[pullup:always]
```

crsctl status type

Use the `crsctl status type` command to obtain the configuration information of one or more particular resource types.

Syntax

```
crsctl status type [resource_type_name [...] | -w "filter"] [-g] [-p] [-f]
```

Parameters

Table E–22 crsctl status type Command Parameters

Parameter	Description
`resource_type_name [...] | -w "filter"`	Specify one or more space-delimited resource type names of which you want to check the status. Alternatively, you can specify a resource type filter surrounded by double quotation marks ("") that Oracle Clusterware uses to match resource types. For example, `-w "TYPE = ora.database.type"`. **See Also:** "Filters" on page E-2 for more information
`[-g] [-p] [-f]`	You can specify the following parameters as options when Oracle Clusterware checks the status of specific server pools: ■ `-g`: Use this parameter to check if the specified resource types are registered ■ `-p`: Use this parameter to display static configuration of the specified resource types ■ `-f`: Use this parameter to display the full configuration of the resource types

Usage Notes

- The *resource_type_name* parameter or a filter is required

Example

The `crsctl status type` command returns output similar to the following:

```
$ crsctl status type ora.network.type

TYPE_NAME=ora.network.type
BASE_TYPE=ora.local_resource.type
```

crsctl stop resource

Use the `crsctl stop resource` command to stop running resources.

Syntax

```
crsctl stop resource {resource_name [...] | -w "filter" | -all} [-n server_name]
   [-k cid] [-d did] [-env "env1=val1,env2=val2,..."]
   [-begin | -end] [-i] [-f] [-l]
```

Parameters

Table E–23 crsctl stop resource Command Parameters

Parameter	Description
`resource_name [...]`	One or more space-delimited resource names to stop.

Table E–23 (Cont.) crsctl stop resource Command Parameters

Parameter	Description
`-w "filter"`	Specify a resource filter that Oracle Clusterware uses to limit the number of resources stopped. The filter must be enclosed in double quotation marks (`""`). Examples of resource filters include: ■ `"TYPE == cluster_resource"`: This filter limits Oracle Clusterware to stop only resources of `cluster_resource` type ■ `"CHECK_INTERVAL > 10"`: This filter limits Oracle Clusterware to stop resources that have a value greater than 10 for the `CHECK_INTERVAL` resource attribute ■ `"(CHECK_INTERVAL > 10) AND (NAME co 2)"`: This filter limits Oracle Clusterware to stop resources that have a value greater than 10 for the `CHECK_INTERVAL` resource attribute *and* the name of the resource contains the number `2` **See Also:** "Filters" on page E-2 for more information
`-all`	Use this parameter to stop all resources in the cluster.
`-n server_name`	Specify the name of the server on which the resource instances you want to stop reside. If you do not specify a server, then Oracle Clusterware stops all instances of the resource.
`-k cid`	Specify the resource cardinality ID. If you specify this parameter, then Oracle Clusterware stops the resource instances that have the cardinality you specify.
`-d did`	Specify the resource degree ID. If you specify this parameter and the degree ID is greater than 1, then Oracle Clusterware stops all resource instances that meet this criteria.
`-env "env1=val1, env2=val2,..."`	You can optionally override one or more resource profile attribute values with the `-env` command parameter. If you specify multiple environment name-value pairs, then you must separate each pair with a comma and enclose the entire list in double quotation marks (`""`).
`-begin`	You can specify this parameter to begin a transparent HA action.
`-end`	You can specify this parameter to end a transparent HA action.
`-i`	If you specify `-i`, then the command returns an error if processing this command requires waiting for Oracle Clusterware to unlock the resource or its dependents. Sometimes, Oracle Clusterware locks resources or other objects to prevent commands from interfering with each other.
`-f`	Specify the `-f` parameter to force the stopping of the resource when it has other resources running that depend on it. Dependent resources are relocated or stopped when you use this parameter.
`-l`	Use the `-l` parameter to leave the resources in the state they were in if the stop command fails.

Usage Notes

- Any one of the three following options is required to specify which resources you want to stop:

  - You can specify one or more resources to stop
  - You can specify a resource filter that Oracle Clusterware uses to match resources to stop
  - You can specify the `-all` parameter with the `-n` *`server_name`* parameter to stop all resources on a particular server
- You must have read and execute permissions on the specified resources to stop them
- Do not use this command for any resources with names that begin with *ora* because these resources are Oracle resources
- Oracle does not support stopping managed applications outside of the Oracle Grid Infrastructure

Example

To stop a resource:

```
$ crsctl stop resource -n node1 -k 2
```

Oracle RAC Environment CRSCTL Commands

The commands listed in this section manage the Oracle Clusterware stack in an Oracle RAC environment, which consists of the following:

- Oracle Clusterware, the member nodes and server pools
- Oracle ASM (if installed)
- Cluster Synchronization Services
- Cluster Time Synchronization Services

You can use the following commands only in an Oracle RAC environment:

- `crsctl add category`
- `crsctl add crs administrator`
- `crsctl add css votedisk`
- `crsctl add policy`
- `crsctl add serverpool`
- `crsctl check cluster`
- `crsctl check crs`
- `crsctl check resource`
- `crsctl check ctss`
- `crsctl config crs`
- `crsctl create policyset`
- `crsctl delete category`
- `crsctl delete crs administrator`
- `crsctl delete css votedisk`
- `crsctl delete node`
- `crsctl delete policy`

- crsctl delete serverpool
- crsctl disable crs
- crsctl discover dhcp
- crsctl enable crs
- crsctl eval activate policy
- crsctl eval add server
- crsctl eval add serverpool
- crsctl eval delete server
- crsctl eval delete serverpool
- crsctl eval modify serverpool
- crsctl eval relocate server
- crsctl get clientid dhcp
- crsctl get cluster hubsize
- crsctl get cluster mode
- crsctl get cpu equivalency
- crsctl get css
- crsctl get css ipmiaddr
- crsctl get css leafmisscount
- crsctl get node role
- crsctl get nodename
- crsctl get resource use
- crsctl get server label
- crsctl getperm serverpool
- crsctl lsmodules
- crsctl modify category
- crsctl modify policy
- crsctl modify policyset
- crsctl modify serverpool
- crsctl pin css
- crsctl query crs administrator
- crsctl query crs activeversion
- crsctl query crs autostart
- crsctl query crs releasepatch
- crsctl query crs releaseversion
- crsctl query crs softwarepatch
- crsctl query crs softwareversion
- crsctl query css ipmiconfig

- crsctl query css ipmidevice
- crsctl query css votedisk
- crsctl query dns
- crsctl query socket udp
- crsctl release dhcp
- crsctl relocate resource
- crsctl relocate server
- crsctl replace discoverystring
- crsctl replace votedisk
- crsctl request action
- crsctl request dhcp
- crsctl set cluster hubsize
- crsctl set cluster mode
- crsctl set cpu equivalency
- crsctl set crs autostart
- crsctl set css
- crsctl set css ipmiaddr
- crsctl set css ipmiadmin
- crsctl set css leafmisscount
- crsctl set node role
- crsctl set resource use
- crsctl set server label
- crsctl setperm serverpool
- crsctl start cluster
- crsctl start crs
- crsctl start ip
- crsctl start rollingpatch
- crsctl start rollingupgrade
- crsctl start testdns
- crsctl status category
- crsctl status ip
- crsctl status policy
- crsctl status policyset
- crsctl status server
- crsctl status serverpool
- crsctl status testdns
- crsctl stop cluster

- crsctl stop crs
- crsctl stop ip
- crsctl stop rollingpatch
- crsctl stop testdns
- crsctl unpin css
- crsctl unset css
- crsctl unset css ipmiconfig
- crsctl unset css leafmisscount

crsctl add category

Use the `crsctl add category` command to add a server category.

Syntax

```
crsctl add category category_name [-attr "attr_name=attr_value
   [,attr_name=attr_value[,...]]"] [-i]
```

Parameters

Table E–24 crsctl add category Command Parameters

Parameter	Description
category_name	Specify a name for the server category you want to add.
attr_name	Specify the name of a category attribute you want to add preceded by the `-attr` flag.
attr_value	A value for the category attribute. **Note:** The *attr_name* and *attr_value* parameters must be enclosed in double quotation marks ("") and separated by commas. For example: `-attr "EXPRESSION='(CPU_COUNT > 2) AND (MEMORY_SIZE > 2048)'"` **See Also:** ■ Table 3–4, " Server Configuration Attributes" for a list of attributes you can use with this command ■ "Filters" on page E-2 for information about operators
-i	If you specify `-i`, then the command fails if Oracle Clusterware cannot process the request immediately.

Usage Notes

- If an attribute value for an attribute name-value pair contains commas, then the value must be enclosed in single quotation marks (''). For example:

  ```
  "START_DEPENDENCIES='hard(res1,res2,res3)'"
  ```

Example

To add a server category:

```
$ crsctl add category cat1 -attr "EXPRESSION='(CPU_COUNT > 2) AND (MEMORY_SIZE >
2048)'"
```

crsctl add crs administrator

Use the `crsctl add crs administrator` command to add a user to the list of cluster administrators.

Syntax

```
crsctl add crs administrator -u user_name [-f]
```

Parameters

Table E–25 crsctl add crs administrator Command Parameters

Parameter	Description
`-u` *user_name*	The name of the user to whom you want to give Oracle Clusterware administrative privileges.
`-f`	Use this parameter to override the user name validity check.

Usage Notes

- This command is deprecated in Oracle Clusterware 12*c*.
- You must run this command as `root` or a cluster administrator, or an administrator on Windows systems
- By default, `root`, the user that installed Oracle Clusterware, and the `*` wildcard are members of the list of users who have Oracle Clusterware administrative privileges. Run the `crsctl delete crs administrator` command to remove the wildcard and enable role-separated management of Oracle Clusterware.

> **See Also:** "Role-Separated Management" on page 2-1 for more information

Example

To add a user to the list of Oracle Clusterware administrators:

```
# crsctl add crs administrator -u scott
```

crsctl add css votedisk

Use the `crsctl add css votedisk` command to add one or more voting files to the cluster on storage devices other than an Oracle ASM disk group.

Syntax

```
crsctl add css votedisk path_to_voting_disk [path_to_voting_disk ...] [-purge]
```

Parameters

Table E–26 crsctl add css votedisk Command Parameters

Parameter	Description
path_to_voting_disk	A fully qualified path to the voting file you want to add. To add multiple voting files, separate each path with a space.
`-purge`	Removes all existing voting files at once. You can replace the existing set of voting files in one operation.

Usage Notes

- You should have at least three voting files, unless you have a storage device, such as a disk array, that provides external redundancy. Oracle recommends that you do not use more than 5 voting files. The maximum number of voting files that is supported is 15.

> **See Also:** "Adding, Deleting, or Migrating Voting Files" on page 6-26 for more information

Example

To add a voting file to the cluster:

```
$ crsctl add css votedisk /stor/grid/ -purge
```

crsctl add policy

Use the `crsctl add policy` command to add a configuration policy to the policy set.

Syntax

```
crsctl add policy policy_name -attr "attr_name=attr_value[,attr_name=attr_value[,
...]]" [-i]
```

Parameters

Table E–27 crsctl add policy Command Parameters

Parameter	Description
policy_name	Specify a name for the policy you want to add.
attr_name	Specify a description for the policy using the `DESCRIPTION` policy attribute preceded by the `-attr` flag.
attr_value	A value for the `DESCRIPTION` policy attribute that describes the policy. **Note:** The *attr_name* and *attr_value* parameters must be enclosed in double quotation marks (`""`) and separated by commas. For example: `-attr "DESCRIPTION=daytime"`
`-i`	If you specify `-i`, then the command fails if Oracle Clusterware cannot process the request immediately.

Usage Notes

- Adding a policy does not activate the policy
- The *policy_name* parameter is required
- Privileges necessary to run this command depend on the value of the `ACL` attribute of the policy set

Example

To add a policy:

```
$ crsctl add policy nightTime -attr "DESCRIPTION=nighttime"
```

crsctl add serverpool

Use the `crsctl add serverpool` command to add a server pool that is for hosting non-database resources (such as application servers) to Oracle Clusterware.

Syntax

```
crsctl add serverpool server_pool_name {-file file_path |
    -attr "attr_name=attr_value[,attr_name=attr_value[,...]]"} [-i] [-f]
```

Parameters

Table E–28 crsctl add serverpool Command Parameters

Parameter	Description
`server_pool_name`	A short, descriptive name for the server pool. A server pool name has a 254 character limit and can contain any platform-supported characters except the exclamation point (!), the tilde (~), and spaces. A server pool name cannot begin with a period nor with *ora*.
`-file` *`file_path`*	Fully-qualified path to an attribute file to define the server pool.
`attribute_name`	The name of a server pool attribute Oracle Clusterware uses to manage the server pool preceded by the `-attr` flag. The available attribute names include: ■ `IMPORTANCE` ■ `MIN_SIZE` ■ `MAX_SIZE` ■ `SERVER_NAMES` ■ `PARENT_POOLS` ■ `EXCLUSIVE_POOLS` ■ `ACL` ■ `SERVER_CATEGORY`
`attribute_value`	A value for the server pool attribute. **Note:** The *`attribute_name`* and *`attribute_value`* parameters must be enclosed in double quotation marks (`""`) and separated by commas. For example: `-attr "MAX_SIZE=30, IMPORTANCE=3"`
`-i`	If you specify `-i`, then the command fails if Oracle Clusterware cannot process the request immediately.
`-f`	If you specify the `-f` parameter, then Oracle Clusterware stops resources running on a server in another server pool and relocates that server into the server pool you are adding. If you do not specify the `-f` parameter, then Oracle Clusterware checks whether the creation of the server pool results in stopping any resources on a server in another server pool that is going to give up a server to the server pool you are adding. If so, then Oracle Clusterware rejects the `crsctl add serverpool` command.

See Also: "How Server Pools Work" on page 3-3 for more information about server pools and server pool attributes

Usage Notes

- The *server_pool_name* parameter is required.
- If an attribute value for an attribute name-value pair contains commas, then the value must be enclosed in single quotation marks ('').
- Do not use this command for any server pools with names that begin with *ora* because these server pools are Oracle server pools.
- Running this command may result in Oracle Clusterware relocating other servers between server pools to comply with the new configuration.
- You must run this command as root or a cluster administrator.
- Use the crsctl add serverpool command to create server pools that host non-database resources. To create server pools that host Oracle databases, use the SRVCTL command utility.

> **See Also:** *Oracle Real Application Clusters Administration and Deployment Guide* for information about using the SRVCTL command utility to create server pools

Examples

Example 1

To add a server pool named testsp with a maximum size of 5 servers, run the following command as root or the Oracle Clusterware installation owner:

```
# crsctl add serverpool testsp -attr "MAX_SIZE=5"
```

Example 2

Create the sp1_attr file with the attribute values for the sp1 serverpool, each on its own line, as shown in the following example:

```
IMPORTANCE=1
MIN_SIZE=1
MAX_SIZE=2
SERVER_NAMES=node3 node4 node5
PARENT_POOLS=Generic
EXCLUSIVE_POOLS=testsp
ACL=owner:oracle:rwx,pgrp:oinstall:rwx,other::r--
```

Use the following command to create the sp1 server pool using the sp1_attr file as input:

```
$ crsctl add serverpool sp1 -file /tmp/sp1_attr
```

crsctl check cluster

Use the crsctl check cluster command on any node in the cluster to check the status of the Oracle Clusterware stack.

Syntax

```
crsctl check cluster [-all | [-n server_name [...]]
```

Usage Notes

- You can check the status of the Oracle Clusterware stack on all nodes in the cluster with the -all parameter or you can specify one or more space-delimited nodes. If you do not specify either parameter, then Oracle Clusterware checks the status of the Oracle Clusterware stack on the local server.

- You can use this cluster-aware command on any node in the cluster.

Example

The `crsctl check cluster` command returns output similar to the following:

```
$ crsctl check cluster -all
*****************************************************************
node1:
CRS-4537: Cluster Ready Services is online
CRS-4529: Cluster Synchronization Services is online
CRS-4533: Event Manager is online
*****************************************************************
node2:
CRS-4537: Cluster Ready Services is online
CRS-4529: Cluster Synchronization Services is online
CRS-4533: Event Manager is online
*****************************************************************
```

crsctl check crs

Use the `crsctl check crs` command to check the status of Oracle High Availability Services and the Oracle Clusterware stack on the local server.

Syntax

```
crsctl check crs
```

Example

To check the health of Oracle Clusterware on the local server:

```
$ crsctl check crs
CRS-4638: Oracle High Availablity Services is online
CRS-4537: Cluster Ready Services is onlin
CRS-4529: Cluster Synchronization Services is online
CRS-4533: Event Manager is online
```

crsctl check resource

Use the `crsctl check resource` command to initiate the check action inside the application-specific agent of a particular resource. Oracle Clusterware only provides output if something prevents the system from issuing the check request, such as a bad resource name.

Syntax

```
crsctl check resource {resource_name [...] | -w "filter" }
      [-n node_name] [-k cardinality_id] [-d degree_id] }
```

Parameters

Table E–29 crsctl check resource Command Parameters

Parameter	Description
resource_name	Specify a particular resource. You can check multiple resources by entering multiple resource names, with each name separated by a space.
-w "*filter*"	Specify a resource filter that Oracle Clusterware uses to limit the number of resources checked. The filter must be enclosed in double quotation marks (""). Examples of resource filters include: ■ "TYPE == cluster_resource": This filter limits Oracle Clusterware to check only resources of cluster_resource type ■ "CHECK_INTERVAL > 10": This filter limits Oracle Clusterware to check resources that have a value greater than 10 for the CHECK_INTERVAL resource attribute ■ "(CHECK_INTERVAL > 10) AND (NAME co 2)": This filter limits Oracle Clusterware to check resources that have a value greater than 10 for the CHECK_INTERVAL resource attribute *and* the name of the resource contains the number 2 **See Also:** "Filters" on page E-2 for more information
-n *node_name*	Check the resource instance on a specific node. If you do not specify the -n parameter, then Oracle Clusterware checks the resource instances only on the local server.
-k *cardinality_id*	Specify the resource cardinality ID.
-d *degree_id*	Specify the resource degree ID.

Usage Notes

- You must have read and execute permissions on the specified resources to check them
- Do not use this command for any resources with names that begin with *ora* because these resources are Oracle resources
- If this command is successful, it only means that a check was issued; it does not mean the CHECK action has been completed

Example

To initiate the check action inside the application-specific agent of a particular resource:

```
$ crsctl check resource appsvip
```

crsctl check ctss

Use the `crsctl check ctss` command to check the status of the Cluster Time Synchronization services.

Syntax

```
crsctl check ctss
```

Example

The `crsctl check ctss` command returns output similar to the following:

```
CRS-4700: The Cluster Time Synchronization Service is in Observer mode.
```

or

```
CRS-4701: The Cluster Time Synchronization Service is in Active mode.
CRS-4702: Offset from the reference node (in msec): 100
```

crsctl config crs

Use the `crsctl config crs` command to display Oracle High Availability Services automatic startup configuration.

Syntax

```
crsctl config crs
```

Example

The `crsctl config crs` command returns output similar to the following:

```
CRS-4622: Oracle High Availability Services autostart is enabled.
```

crsctl create policyset

Use the `crsctl create policyset` command to create a single policy set, in the form of a text file, that reflects the server pool configuration. After you create a policy set, you can copy the contents of the text file to create other policy sets.

Syntax

```
crsctl create policyset -file path_to_file
```

Parameters

Table E–30 crsctl create policyset Command Parameters

Parameter	Description
`-file` *`file_name`*	Specify a path to where CRSCTL creates a file that you can edit and then send back using `crsctl modify policyset` to add, delete, or update policies.

Example

To create a policy set:

```
$ crsctl create policyset -file /tmp/ps
```

crsctl delete category

Use the `crsctl delete category` command to delete a server category.

Syntax

```
crsctl delete category category_name [category_name [...]] [-i]
```

Parameters

Table E–31 crsctl delete category Command Parameters

Parameter	Description
category_name	Specify the name of the server category or a space-delimited list of server categories that you want to delete.
-i	If you specify -i, then the command fails if Oracle Clusterware cannot process the request immediately.

Example

To delete a server category:

```
$ crsctl delete category blue_server -i
```

crsctl delete crs administrator

Use the `crsctl delete crs administrator` command to remove a user from the Oracle Clusterware administrator list.

Syntax

```
crsctl delete crs administrator -u user_name [-f]
```

Parameters

Table E–32 crsctl delete crs administrator Command Parameters

Parameter	Description
-u *user_name*	The name of the user whose Oracle Clusterware administrative privileges you want to remove. By default, the list of users with Oracle Clusterware administrative privileges consists of the user who installed Oracle Clusterware, root, and *. The user who installed Oracle Clusterware and root are permanent members this list. The * value gives Oracle Clusterware administrative privileges to all users and must be removed to enable role-separated management. **See Also:** "Role-Separated Management" on page 2-1 for more information
-f	Use this parameter to override the user name validity check.

Usage Notes

- The *user_name* parameter is required
- You must run this command as root or a cluster administrator, or an administrator on Windows systems
- To enable role-separated management, you must remove the * value enclosed in double quotation marks ("")

Example

To remove a user from the list of cluster administrators:

```
# crsctl delete crs administrator -u scott
```

crsctl delete css votedisk

Use the `crsctl delete css votedisk` to remove a voting file from the Oracle Clusterware configuration.

Syntax

```
crsctl delete css votedisk {voting_disk_GUID [...] | vdisk [...] | +diskgroup}
```

Parameters

Table E–33 crsctl delete css votedisk Command Parameters

Parameter	Description
voting_disk_GUID	Enter the file universal identifier (GUID) of the voting file you want to remove. Specify multiple GUIDs in a space-delimited list.
vdisk	Enter the path of the voting file you want to remove. Specify multiple voting file paths in a space-delimited list.
+*diskgroup*	Enter the name of an Oracle ASM disk group that contains voting files you want to remove. You can only use parameter when Oracle Clusterware is in exclusive mode.

Usage Notes

- You can specify one or more GUIDs of voting files you want to remove, one or paths to voting files you want to remove, or the name of an Oracle ASM disk group that contains voting files you want to remove.
- You can obtain the GUIDs of each current voting file by running the `crsctl query css votedisk` command

Example

To remove a voting file:

```
$ crsctl delete css votedisk 26f7271ca8b34fd0bfcdc2031805581e
```

crsctl delete node

Use the `crsctl delete node` to remove a node from the cluster.

Syntax

```
crsctl delete node -n node_name
```

Usage Notes

- You must be `root` to run this command
- The *node_name* parameter is required
- You cannot use this command on a Leaf Node in an Oracle Flex Cluster

Example

To delete the node named `node06` from the cluster, run the following command as `root`:

```
# crsctl delete node -n node06
```

crsctl delete policy

Use the `crsctl delete policy` command to delete a configuration policy from the policy set.

Syntax

```
crsctl delete policy policy_name [policy_name [...]] [-i]
```

Parameters

Table E–34 crsctl delete policy Command Parameters

Parameter	Description
policy_name	Specify a name for the policy or a space-delimited list of policy names you want to delete.
-i	If you specify `-i`, then the command fails if Oracle Clusterware cannot process the request immediately.

Usage Notes

- The *policy_name* parameter is required
- Privileges necessary to run this command depend on the value of the `ACL` attribute of the policy set

Example

To delete a policy, run the following command as `root` or the Oracle Clusterware installation owner:

```
# crsctl delete policy...
```

crsctl delete serverpool

Use the `crsctl delete serverpool` command to remove a server pool from the Oracle Clusterware configuration.

Syntax

```
crsctl delete serverpool server_pool_name [server_pool_name [...]] [-i]
```

> **See Also:** "How Server Pools Work" on page 3-3 for more information about server pools and server pool attributes

Usage Notes

- The *server_pool_name* parameter is required
- If you specify `-i`, then the command fails if Oracle Clusterware cannot process the request immediately
- Do not use this command for any server pools with names that begin with *ora* because these server pools are Oracle server pools
- While you can use this command in either environment, it is only useful in the Oracle RAC environment

Example

To delete a server pool, run the following command as root or the Oracle Clusterware installation owner:

```
# crsctl delete serverpool sp1
```

crsctl disable crs

Use the crsctl disable crs command to prevent the automatic startup of Oracle High Availability Services when the server boots.

Syntax

```
crsctl disable crs
```

Usage Notes

- This command only affects the local server
- If you disable Oracle High Availability Services automatic startup, you must use the crsctl start crs command to start Oracle High Availability Services

Example

The crsctl disable crs command returns output similar to the following:

```
CRS-4621: Oracle High Availability Services autostart is disabled.
```

crsctl discover dhcp

Use the crsctl discover dhcp command to send DHCP discover packets on the network at the specified port. If DHCP servers are present on the network, then they respond to the discovery message and the command succeeds.

Syntax

```
crsctl discover dhcp -clientid clientid [-port port]
```

Parameters

Table E–35 crsctl discover dhcp Command Parameters

Parameter	Description
-clientid *clientid*	Specify the client ID for which you want to attempt discovery. Obtain the client ID by running the crsctl get clientid dhcp command.
-port *port*	The port to which CRSCTL sends the discovery packets.

Usage Notes

You must run this command as root

Example

The crsctl discover dhcp command returns output similar to the following:

```
# crsctl discover dhcp -clientid stmjk0462clr-stmjk01-vip

CRS-10009: DHCP server returned server: 192.168.53.232,
 loan address : 192.168.29.221/255.255.252.0, lease time: 43200
```

crsctl enable crs

Use the `crsctl enable crs` command to enable automatic startup of Oracle High Availability Services when the server boots.

Syntax

```
crsctl enable crs
```

Usage Notes

- This command only affects the local server

Example

The `crsctl enable crs` command returns output similar to the following:

```
CRS-4622: Oracle High Availability Services autostart is enabled.
```

crsctl eval activate policy

Use the `crsctl eval activate policy` command to predict the effects of activating a specific policy without making changes to the system. This command may be useful to cluster administrators.

Syntax

```
crsctl eval activate policy policy_name [-f] [-admin [-l serverpools | resources
  | all] [-x] [-a]]
```

Parameters

Table E–36 crsctl eval activate policy Command Parameters

Parameter	Description
`-f`	Specify this parameter to evaluate what happens if you try to forcibly activate a policy.
`-admin [-l serverpools	resources	all] [-x] [-a]`	You must specify `-admin` if you specify any combination of `-l`, `-x`, and `-a`. If you specify the `-l` parameter, then you can choose one of the following three output levels: - `serverpools`: Restricts the output to servers running in a server pool - `resources`: Restricts the output to resources running on servers in a server pool - `all`: Displays all available output If you specify the `-x` parameter, then CRSCTL displays the differences. If you specify the `-a` parameter, then CRSCTL displays all resources.

See Also: "How Server Pools Work" on page 3-3 for more information about server pools and server pool attributes

crsctl eval add server

Use the `crsctl eval add server` command to simulate the addition of a server without making changes to the system. This command may be useful to cluster administrators.

Syntax

```
crsctl eval add server server_name [-file file_path] | [-attr "attr_name=attr_
value[,...]"]
    [-admin [-l level [-x] [-a]] [-f]
```

Parameters

Table E–37 crsctl eval add server Command Parameters

Parameter	Description
server_name	The name of the server you want to add.
`-file` *file_path*	Fully-qualified path to a file containing server attributes.
attr_name	The name of a server attribute that Oracle Clusterware uses to manage the server preceded by the `-attr` flag. **See Also:** "Server Category Attributes" on page 3-14 for information about server attributes
attr_value	A value for the server attribute. **Note:** The *attribute_name* and *attribute_value* parameters must be enclosed in double quotation marks (`""`) and separated by commas. For example: `-attr "MAX_SIZE=30,IMPORTANCE=3"`
`-admin [-l` *level*`] [-x] [-a]`	If you specify this parameter, then CRSCTL displays output for the cluster administrator. If you specify the `-l` parameter, then you can choose one of the following three output levels: ■ `serverpools`: Restricts the output to servers running in a server pool ■ `resources`: Restricts the output to resources running on servers in a server pool ■ `all`: Displays all available output If you specify the `-x` parameter, then CRSCTL displays the differences. If you specify the `-a` parameter, then CRSCTL displays all resources. **Note:** To specify either the `-l`, `-x`, or `-a` parameters, or any combination of the three, you must specify the `-admin` parameter.
`-f`	If you specify this parameter, then CRSCTL predicts the effects of forcibly adding a server.

> **See Also:** Chapter 7, "Adding and Deleting Cluster Nodes" for more information about adding servers

Example

The following example predicts how the system reacts when you add a server called `mjkeenan-node-3`:

```
# crsctl eval add server mjkeenan-node-3 -admin -l resources -a

--------------------------------------------------------------------------------
Name           Target  State        Server                   Effect
--------------------------------------------------------------------------------
Local Resources
--------------------------------------------------------------------------------
ora.net1.network
               ONLINE  ONLINE       mjkeenan-node-0
               ONLINE  ONLINE       mjkeenan-node-1
--------------------------------------------------------------------------------
Cluster Resources
--------------------------------------------------------------------------------
cs1
      1        ONLINE  ONLINE       mjkeenan-node-1
      2        ONLINE  ONLINE       mjkeenan-node-0
cs2
      1        ONLINE  ONLINE       mjkeenan-node-3          Started
ora.gns
```

crsctl eval add serverpool

Use the `crsctl eval add serverpool` command to predict the effects of adding a server pool without making changes to the system. This command may be useful to cluster administrators.

> **See Also:** "How Server Pools Work" on page 3-3 for more information about server pools and server pool attributes

Syntax

```
crsctl eval add serverpool server_pool_name [-file file_path | -attr "attr_
name=attr_value
    [,attr_name=attr_value[,...]]" [-admin [-l level [-x] [-a]] [-f]
```

Parameters

Table E–38 crsctl eval add serverpool Command Parameters

Parameter	Description
server_pool_name	A short, descriptive name for the server pool.
-file *file_path*	Fully-qualified path to an attribute file to define the server pool.
attribute_name	The name of a server pool attribute Oracle Clusterware uses to manage the server pool preceded by the `-attr` flag. The available attribute names include: ▪ IMPORTANCE ▪ MIN_SIZE ▪ MAX_SIZE ▪ SERVER_NAMES ▪ PARENT_POOLS ▪ EXCLUSIVE_POOLS ▪ ACL

Table E–38 (Cont.) crsctl eval add serverpool Command Parameters

Parameter	Description
`attribute_value`	A value for the server pool attribute. **Note:** The `attribute_name` and `attribute_value` parameters must be enclosed in double quotation marks (`""`) and separated by commas. For example: `-attr "MAX_SIZE=30,IMPORTANCE=3"`
`-admin [-l level] [-x] [-a]`	If you specify this parameter, then CRSCTL displays output for the cluster administrator. If you specify the `-l` parameter, then you can choose one of the following three output levels: ▪ `serverpools`: Restricts the output to servers running in a server pool ▪ `resources`: Restricts the output to resources running on servers in a server pool ▪ `all`: Displays all available output If you specify the `-x` parameter, then CRSCTL displays the differences. If you specify the `-a` parameter, then CRSCTL displays all resources. **Note:** To specify either the `-l`, `-x`, or `-a` parameters, or any combination of the three, you must specify the `-admin` parameter.
`-f`	If you specify this parameter, then CRSCTL predicts the effects of forcibly adding a server pool.

crsctl eval delete server

Use the `crsctl eval delete server` command to predict the effects of deleting a server without making changes to the system. This command may be useful to cluster administrators.

> **See Also:** "How Server Pools Work" on page 3-3 for more information about server pools and server pool attributes

Syntax

```
crsctl eval delete server server_name [-admin [-l level] [-x] [-a]] [-f]
```

Parameters

Table E–39 crsctl eval delete server Command Parameters

Parameter	Description
`server_name`	Specify the name of the server you want to evaluate before deleting.

Table E–39 (Cont.) crsctl eval delete server Command Parameters

Parameter	Description
`-admin [-l level] [-x] [-a]`	If you specify this parameter, then CRSCTL displays output for the cluster administrator. If you specify the `-l` parameter, then you can choose one of the following three output levels: ■ `serverpools`: Restricts the output to servers running in a server pool ■ `resources`: Restricts the output to resources running on servers in a server pool ■ `all`: Displays all available output If you specify the `-x` parameter, then CRSCTL displays the differences. If you specify the `-a` parameter, then CRSCTL displays all resources. **Note:** To specify either the `-l`, `-x`, or `-a` parameters, or any combination of the three, you must specify the `-admin` parameter.

crsctl eval delete serverpool

Use the `crsctl eval delete serverpool` command to simulate the deletion of a server pool without making changes to the system. This command may be useful to cluster administrators.

> **See Also:** "How Server Pools Work" on page 3-3 for more information about server pools and server pool attributes

Syntax

```
crsctl eval delete serverpool server_pool_name [-admin [-l level] [-x] [-a]]
```

Parameters

Table E–40 crsctl eval delete serverpool Command Parameters

Parameter	Description
server_pool_name	The name of the server pool you want to delete.
`-admin [-l level] [-x] [-a]`	If you specify this parameter, then CRSCTL displays output for the cluster administrator. If you specify the `-l` parameter, then you can choose one of the following three output levels: ■ `serverpools`: Restricts the output to servers running in a server pool ■ `resources`: Restricts the output to resources running on servers in a server pool ■ `all`: Displays all available output If you specify the `-x` parameter, then CRSCTL displays the differences. If you specify the `-a` parameter, then CRSCTL displays all resources. **Note:** To specify either the `-l`, `-x`, or `-a` parameters, or any combination of the three, you must specify the `-admin` parameter.

crsctl eval modify serverpool

Use the `crsctl eval modify serverpool` command to predict the effects of modifying a server pool without making changes to the system. This command may be useful to cluster administrators.

Syntax

```
crsctl eval modify serverpool server_pool_name {-file file_path
   | -attr "attr_name=attr_value [,attr_name=attr_value[, ...]]"}
   [-f] [-admin [-l level [-x] [-a]]
```

Parameters

> **See Also:** "crsctl modify serverpool" on page E-66 for a description of the `-attr` parameter

Table E–41 crsctl eval modify serverpool Command Parameters

Parameter	Description
server_pool_name	The name of the server pool you want to modify.
`-f`	If you specify this parameter, then CRSCTL predicts the effects of forcibly modifying a server pool.
`-admin [-l` *level*`] [-x] [-a]`	If you specify this parameter, then CRSCTL displays output for the cluster administrator. If you specify the `-l` parameter, then you can choose one of the following three output levels: ■ `serverpools`: Restricts the output to servers running in a server pool ■ `resources`: Restricts the output to resources running on servers in a server pool ■ `all`: Displays all available output If you specify the `-x` parameter, then CRSCTL displays the differences. If you specify the `-a` parameter, then CRSCTL displays all resources. **Note:** To specify either the `-l`, `-x`, or `-a` parameters, or any combination of the three, you must specify the `-admin` parameter.

> **See Also:** "How Server Pools Work" on page 3-3 for more information about server pools and server pool attributes

Usage Notes

- The *server_pool_name* parameter is required
- If an attribute value for an attribute name-value pair contains commas, then the value must be enclosed in single quotation marks (`''`). For example:

  ```
  "START_DEPENDENCIES='hard(res1,res2,res3)'"
  ```

- Running this command may result in Oracle Clusterware relocating other servers between server pools to comply with the new configuration

- Do not use this command for any server pools with names that begin with *ora* because these server pools are Oracle server pools
- While you can use this command in either environment, it is only useful in the Oracle RAC environment

crsctl eval relocate server

Use the `crsctl eval relocate server` command to predict the effects of relocating a server to a different server pool without making changes to the system. This command might be useful for a cluster administrator.

Syntax

```
crsctl eval relocate server server_name -to server_pool_name [-f]
[-admin [-l level] [-x] [-a]]
```

Parameters

Table E–42 crsctl eval relocate server Command Parameters

Parameter	Description
server_name	The name of the server you want to relocate. You can provide a space-delimited list of servers to evaluate relocating multiple servers.
`-to`	Specify the name of the server pool to which you want relocate the server.
`-f`	If you specify this parameter, then CRSCTL predicts the effects of forcibly relocating a server.
`-admin [-l` *level*`] [-x] [-a]`	If you specify this parameter, then CRSCTL displays output for the cluster administrator. If you specify the `-l` parameter, then you can choose one of the following three output levels: ■ `serverpools`: Restricts the output to servers running in a server pool ■ `resources`: Restricts the output to resources running on servers in a server pool ■ `all`: Displays all available output If you specify the `-x` parameter, then CRSCTL displays the differences. If you specify the `-a` parameter, then CRSCTL displays all resources. **Note:** To specify either the `-l`, `-x`, or `-a` parameters, or any combination of the three, you must specify the `-admin` parameter.

crsctl get clientid dhcp

Use the `crsctl get clientid dhcp` command to display the client ID that the Oracle Clusterware agent uses to obtain the IP addresses from the DHCP server for configured cluster resources. The VIP type is required.

Syntax

```
crsctl get clientid dhcp -cluname cluster_name -viptype vip_type
[-vip vip_res_name] [-n node_name]
```

Parameters

Table E–43 crsctl get clientid dhcp Command Parameters

Parameter	Description
`-cluname` *cluster_name*	Specify the name of the cluster where the cluster resources are configured.
`-viptype` *vip_type*	Specify the type of the VIP resource for which you want to display client IDs: `HOSTVIP`, `SCANVIP`, or `APPVIP`.
`-vip` *vip_resource_name*	Specify the name of the VIP resource. This parameter is required if you specify the `APPVIP` VIP type.
`-n` *node_name*	Specify the name of the node for which you want to obtain the client ID. This parameter is required if you specify the `HOSTVIP` VIP type.

Example

The `crsctl get clientid dhcp` command returns output similar to the following:

```
$ crsctl get clientid dhcp -cluname stmjk0462clr -viptype HOSTVIP -n stmjk01

CRS-10007: stmjk0462clr-stmjk01-vip
```

crsctl get cluster hubsize

Use the `crsctl get cluster hubsize` command to obtain the value of Hub Nodes in an Oracle Flex Cluster.

Syntax

```
crsctl get cluster hubsize
```

Example

The `crsctl get cluster hubsize` command returns output similar to the following:

```
CRS-4950: Current hubsize parameter value is 32
```

crsctl get cluster mode

Use the `crsctl get cluster mode` command to ascertain whether the cluster is configured for Oracle Flex Clusters or the current status.

Syntax

```
crsctl get cluster mode [config | status]
```

Usage Notes

- Specify the `config` option to obtain the mode in which the cluster is configured.
- Specify the `status` option to obtain the current status of the cluster.

crsctl get cpu equivalency

Use the `crsctl cpu equivalency` command to obtain the value of the `CPU_EQUIVALENCY` server configuration attribute.

Syntax

```
crsctl get cpu equivalency
```

crsctl get css

Use the `crsctl get css` command to obtain the value of a specific Cluster Synchronization Services parameter.

Syntax

```
crsctl get css parameter
```

Usage Notes

- Cluster Synchronization Services parameters and their default values include:

  ```
  clusterguid
  diagwait
  disktimeout (200 (seconds))
  misscount (30 (seconds))
  reboottime (3 (seconds))
  priority (4 (UNIX), 3 (Windows))
  logfilesize (50 (MB))
  ```

- This command does not display default values
- This command only affects the local server

Example

The `crsctl get css disktimeout` command returns output similar to the following:

```
$ crsctl get css disktimeout
CRS-4678: Successful get disktimeout 200 for Cluster Synchronization Services.
```

crsctl get css ipmiaddr

Use the `crsctl get css ipmiaddr` command to get the address stored in the Oracle Local Registry of the local Intelligent Platform Management Interface (IPMI) device.

Syntax

```
crsctl get css ipmiaddr
```

Usage Notes

- Run the command under the user account used to install Oracle Clusterware.
- This command only obtains the IP address stored in the Oracle Local Registry. It may not be the IP address actually used by IPMI.

  Use either `ipmiutil` or `ipmitool` as `root` on the local server to obtain the IP address used by the IPMI device.

Example

To obtain the IPMI device IP address:

```
$ crsctl get css ipmiaddr
```

crsctl get css leafmisscount

Use the `crsctl get css leafmisscount` command to obtain the amount of time (in seconds) that must pass without any communication between a Leaf Node and the Hub Node to which it is attached, before the connection is declared to be no longer active and the Leaf Node is removed from the cluster.

Syntax

```
crsctl get css leafmisscount
```

crsctl get node role

Use the `crsctl get node role` command to obtain the configured node role of nodes in the cluster.

Syntax

```
crsctl get node role {config | status} [node node_name | -all]
```

Usage Notes

- Specify the `config` option to obtain the configured node role for a specific node.
- Specify the `status` option to obtain the current status of a specific node.
- You can specify a particular node for which to obtain role information. If you do not specify a particular node, then CRSCTL returns information about the local node.

Example

The `crsctl get node role` command returns output similar to the following:

```
Node 'adc6140524' configured role is 'hub'
```

crsctl get nodename

Use the `crsctl get nodename` command to obtain the name of the local node.

Syntax

```
crsctl get nodename
```

Example

The `crsctl get nodename` command returns output similar to the following:

```
node2
```

crsctl get resource use

Use the `crsctl get resource use` command to check the current setting value of the `RESOURCE_USE_ENABLED` parameter.

Syntax

```
crsctl get resource use
```

Usage Notes

The possible values are 1 or 0. If the value for this attribute is 1, which is the default, then the server can be used for resource placement. If the value is 0, then Oracle Clusterware disallows starting server pool resources on the server. The server remains in the Free server pool.

Example

This command returns output similar to the following:

```
CRS-4966: Current resource use parameter value is 1
```

crsctl get server label

Use the `crsctl get server label` command to check the current setting value of the `SERVER_LABEL` server attribute.

Syntax

```
crsctl get server label
```

Example

The `crsctl get server label` command returns output similar to the following:

```
CRS-4972: Current SERVER_LABEL parameter value is hubserver
```

crsctl getperm serverpool

Use the `crsctl getperm serverpool` command to obtain permissions for a particular server pool.

Syntax

```
crsctl getperm serverpool server_pool_name  [-u user_name | -g group_name]
```

> **See Also:** "How Server Pools Work" on page 3-3 for more information about server pools and server pool attributes

Parameters

Table E–44 crsctl getperm serverpool Command Parameters

Parameter	Description
server_pool_name	Specify the name of the server pool for which you want to obtain permissions.
-u *user_name*	If you specify -u, then Oracle Clusterware obtains permissions for a particular user.
-g *group_name*	If you specify -g, then Oracle Clusterware obtains permissions for a particular group.

Usage Notes

- The *server_pool_name* parameter is required
- Do not use this command for any server pools with names that begin with *ora* because these server pools are Oracle server pools

- While you can use this command in either environment, it is only useful in the Oracle RAC environment

Example

The `crsctl getperm serverpool` command returns output similar to the following:

```
$ crsctl getperm serverpool sp1
NAME: sp1
owner:root:rwx,pgrp:root:r-x,other::r--
```

crsctl lsmodules

Use the `crsctl lsmodules` command to list the components of the modules that you can debug.

> **See Also:** "Dynamic Debugging Using crsctl set log" on page E-102 for more information about debugging

Syntax

```
crsctl lsmodules {mdns | gpnp | css | crf | crs | ctss | evm | gipc}
```

Usage Notes

You can specify any of the following components:

`mdns`: Multicast domain name server
`gpnp`: Grid Plug and Play service
`css`: Cluster Synchronization Services
`crf`: Cluster Health Monitor
`crs`: Cluster Ready Services
`ctss`: Cluster Time Synchronization Service
`evm`: Event Manager
`gipc`: Grid Interprocess Communication

Example

The `crsctl lsmodules` command returns output similar to the following:

```
$ crsctl lsmodules evm
List EVMD Debug Module: CLSVER
List EVMD Debug Module: CLUCLS
List EVMD Debug Module: COMMCRS
List EVMD Debug Module: COMMNS
List EVMD Debug Module: CRSOCR
List EVMD Debug Module: CSSCLNT
List EVMD Debug Module: EVMAGENT
List EVMD Debug Module: EVMAPP
...
```

crsctl modify category

Use the `crsctl modify category` command to modify an existing server category.

Syntax

```
crsctl modify category category_name [-attr "attr_name=attr_value
   [,attr_name=attr_value[,...]]"] [-i] [-f]
```

Parameters

Table E–45 crsctl modify category Command Parameters

Parameter	Description
category_name	Specify the name of the server category you want to modify.
attr_name	Specify the name of a category attribute you want to modify preceded by the -attr flag.
attr_value	A value for the category attribute. **Note:** The *attr_name* and *attr_value* parameters must be enclosed in double quotation marks ("") and separated by commas. For example: "ACL='owner:st-cdc\cdctest:rwx,pgrp::---', ACTIVE_CSS_ROLE=leaf" **See Also:** ■ Table 3–4, " Server Configuration Attributes" for a list of attributes you can use with this command ■ "Filters" on page E-2 for information about operators
-i	If you specify -i, then the command fails if Oracle Clusterware cannot process the request immediately.
-f	Force parameter

Usage Notes

- If an attribute value for an attribute name-value pair contains commas, then the value must be enclosed in single quotation marks (''). For example:

```
"START_DEPENDENCIESs=s'hard(res1,res2,res3)'"
```

Example

To modify a server category:

```
$ crsctl modify category blue_server -attr  "EXPRESSION=(LOCATION=hub)"
```

crsctl modify policy

Use the `crsctl modify policy` command to modify an existing configuration policy.

Syntax

```
crsctl modify policy policy_name -attr "attr_name=attr_value" [-i]
```

Parameters

Table E–46 crsctl modify policy Command Parameters

Parameter	Description
policy_name	The name of the policy you want to modify.
attr_name	Specify a description for the policy using the DESCRIPTION policy attribute preceded by the -attr flag.

Table E–46 (Cont.) crsctl modify policy Command Parameters

Parameter	Description
attr_value	A value for the `DESCRIPTION` policy attribute that describes the policy. **Note:** The *attr_name* and *attr_value* parameters must be enclosed in double quotation marks ("") and separated by commas. For example: `-attr "DESCRIPTION=daytime"`
`-i`	If you specify `-i`, then the command fails if Oracle Clusterware cannot process the request immediately.

Usage Notes

- The *policy_name* parameter is required
- Privileges necessary to run this command depend on the value of the `ACL` attribute of the policy set

Example

To modify an existing policy, run the following command as `root` or the Oracle Clusterware installation owner:

```
# crsctl modify policy p1 -attr "DESCRIPTION=daytime"
```

crsctl modify policyset

Use the `crsctl modify policyset` command to modify an existing policy set.

Syntax

```
crsctl modify policyset {-attr "attr_name=attr_value[,attr_name=attr_value[,
...]]" | -file file_name} [-ksp]
```

Parameters

Table E–47 crsctl modify policyset Command Parameters

Parameter	Description

attr_name	The name of a policy attribute you want to modify preceded by the `-attr` flag. With this command, you can specify any of the following attributes:
	`ACL`
	`LAST_ACTIVATED_POLICY`
	`SERVER_POOL_NAMES`

attr_value	A value for the policy attribute.
	Note: The *attr_name* and *attr_value* parameters must be enclosed in double quotation marks ("") and separated by commas. For example:
	`-attr`
	`"ACL='owner:mjkeenan:rwx,pgrp:svrtech:rwx,other::r--'`
	`',`
	`SERVER_POOL_NAMES=sp1 sp2 Free"`

`-file` *file_name*	If you specify this parameter instead of `-attr`, then enter a name of a file that contains policy set definitions.

Table E–47 (Cont.) crsctl modify policyset Command Parameters

Parameter	Description
`-ksp`	If you specify this parameter, then CRSCTL keeps the server pools in the system, which means that they are independent and not managed by the policy set.

Usage Notes

- Privileges necessary to run this command depend on the value of the `ACL` attribute of the policy set
- You can only specify policy definitions using the `-file` parameter or by running the `crsctl modify policy` command

Example

To modify an existing policy set, run the following command as `root` or the Oracle Clusterware installation owner:

```
# crsctl modify policyset -file my_policy_set.def
```

crsctl modify server

Use the `crsctl modify server` command to modify the configuration of a particular server.

Syntax

```
crsctl modify server server_name -attr "attr_name=attr_value
   [,attr_name=attr_value[, ...]]"
```

Parameters

Table E–48 crsctl modify server Command Parameters

Parameter	Description
`server_name` `[...]`	Specify the name of the server you want to modify.
`attr_name`	The name of a server attribute you want to modify preceded by the `-attr` flag.
`attr_value`	A value for the server attribute. **Note:** The *`attr_name`* and *`attr_value`* parameters must be enclosed in double quotation marks (`""`) and separated by commas. For example: `-attr "CHECK_INTERVAL=30,START_TIMEOUT=25"`

Usage Notes

- If an attribute value for an attribute name-value pair contains commas, then the value must be enclosed in single quotation marks (`''`). For example:

  ```
  "START_DEPENDENCIES='hard(res1,res2,res3)'"
  ```

Example

To modify the configuration of a specific server, enter a command similar to the following:

```
$ crsctl modify server email_server_2 -attr  "WORKLOAD_VALUE=1"
```

crsctl modify serverpool

Use the `crsctl modify serverpool` command to modify an existing server pool.

Syntax

```
crsctl modify serverpool server_pool_name -attr "attr_name=attr_value
   [,attr_name=attr_value[, ...]]" [-policy policyName | -all_policies]
[-i] [-f]
```

Parameters

Table E–49 crsctl modify serverpool Command Parameters

Parameter	Description
server_pool_name	The name of the server pool you want to modify.
attr_name	The name of a server pool attribute you want to modify preceded by the `-attr` flag. **See Also:** Table 3–1, " Server Pool Attributes" for details about server pool attributes
attr_value	A value for the server pool attribute. **Note:** The *attr_name* and *attr_value* parameters must be enclosed in double quotation marks ("") and separated by commas. For example: `-attr "CHECK_INTERVAL=30,START_TIMEOUT=25"`
`-policy` *policyName* | `-all_policies`	Specify a particular policy or all policies for which you want to modify the server pool definition.
`-i`	If you specify `-i`, then the command fails if Oracle Clusterware cannot process the request immediately.
`-f`	If you specify the `-f` parameter, then Oracle Clusterware stops resources running on a server in another server pool and relocates that server into the server pool you are adding. If you do not specify the `-f` parameter, then Oracle Clusterware checks whether the creation of the server pool results in stopping any resources on a server in another server pool that is going to give up a server to the server pool you are adding. If so, then Oracle Clusterware rejects the `crsctl add serverpool` command.

> **See Also:** "How Server Pools Work" on page 3-3 for more information about server pools and server pool attributes

Usage Notes

- The *server_pool_name* parameter is required
- If an attribute value for an attribute name-value pair contains commas, then the value must be enclosed in single quotation marks (''). For example:

  ```
  "START_DEPENDENCIES='hard(res1,res2,res3)'"
  ```

- Running this command may result in Oracle Clusterware relocating other servers between server pools to comply with the new configuration
- Do not use this command for any server pools with names that begin with *ora* because these server pools are Oracle server pools

- While you can use this command in either environment, it is only useful in the Oracle RAC environment

Example

To modify an existing server pool, run the following command as `root` or the Oracle Clusterware installation owner:

```
# crsctl modify serverpool sp1 -attr "MAX_SIZE=7"
```

crsctl pin css

Use the `crsctl pin css` command to pin many specific nodes. Pinning a node means that the association of a node name with a node number is fixed. If a node is not pinned, its node number may change if the lease expires while it is down. The lease of a pinned node never expires.

Syntax

```
crsctl pin css -n node_name [ node_name [..]]
```

Usage Notes

- You can specify a space-delimited list of servers
- Any pre-12*c* release 1 (12.1) Oracle software must reside on a pinned server.
- A node may be unpinned with `crsctl unpin css`.
- Deleting a node with the `crsctl delete node` command implicitly unpins the node.

Example

To pin the node named `node2`:

```
# crsctl pin css -n node2
```

crsctl query crs administrator

Use the `crsctl query crs administrator` command to display the list of users with Oracle Clusterware administrative privileges.

Syntax

```
crsctl query crs administrator
```

Example

The `crsctl query crs administrator` command returns output similar to the following:

```
CRS Administrator List: scott
```

crsctl query crs activeversion

Use the `crsctl query crs activeversion` command to display the active version and the configured patch level of the Oracle Clusterware software running in the cluster. During a rolling upgrade, however, the active version is not advanced until the upgrade is finished across the cluster, until which time the cluster operates at the pre-upgrade version.

Additionally, during a rolling patch, the active patch level is not advanced until the patching is finished across the cluster, until which time the cluster operates at the pre-upgrade patch level.

Syntax

```
crsctl query crs activeversion [-f]
```

If you specify the `-f` parameter, then this command also prints the patch level for each configured node in the cluster.

Example

The `crsctl query crs activeversion` command returns output similar to the following:

```
$ crsctl query crs activeversion -f

Oracle Clusterware active version on the cluster is [12.1.0.0.2]. The cluster
upgrade state is [NORMAL]. The cluster active patch level is [456789126].
```

crsctl query crs autostart

Use the `crsctl query crs autostart` command to obtain the values of the Oracle Clusterware automatic resource start criteria.

Syntax

```
crsctl query crs autostart
```

Example

The `crsctl query crs autostart` command returns output similar to the following:

```
'Autostart delay':       60
'Autostart servercount': 2
```

crsctl query crs releasepatch

Use the `crsctl query crs releasepatch` command to display the patch level which is updated in the Grid home patch repository while patching a node. The patch level corresponds to only the local node in which the command is executed. This command can be executed while the stack is not running.

Syntax

```
crsctl query crs releasepatch
```

Example

The `crsctl query crs releasepatch` command returns output similar to the following for a node which has no patches applied:

```
Oracle Clusterware release patch level is [3180840333] and the complete list of
patches is [13559647] on the local node.
```

crsctl query crs releaseversion

Use the `crsctl query crs releaseversion` command to display the version of the Oracle Clusterware software stored in the binaries on the local node.

Syntax

```
crsctl query crs releaseversion
```

Example

The `crsctl query crs releaseversion` command returns output similar to the following:

```
Oracle High Availablity Services release version on the local node is [11.2.0.2.0]
```

crsctl query crs softwarepatch

Use the `crsctl query crs softwarepatch` command to display the configured patch level of the installed Oracle Clusterware.

Syntax

```
crsctl query crs softwarepatch [host_name]
```

If you specify a host name, then CRSCTL displays the patch level of Oracle Clusterware installed on that host. Otherwise, CRSCTL displays the patch level of Oracle Clusterware installed on the local host.

Example

The `crsctl query crs softwarepatch` command returns output similar to the following:

```
Oracle Clusterware patch level on node [node1] is [456789126]
```

crsctl query crs softwareversion

Use the `crsctl query crs softwareversion` command to display latest version of the software that has been successfully started on the specified node.

Syntax

```
crsctl query crs softwareversion [node_name]
```

Usage Notes

- If you do not provide a node name, then Oracle Clusterware displays the version of Oracle Clusterware running on the local server.

Example

The `crsctl query crs softwareversion` command returns output similar to the following:

```
Oracle Clusterware version on node [node1] is [11.2.0.2.0]
```

crsctl query css ipmiconfig

Use the `crsctl query css ipmiconfig` command to determine whether Oracle Clusterware on the local server has been configured to use IPMI for failure isolation. Note that this command detects the presence of configuration data, but cannot not validate its correctness.

Syntax

```
crsctl query css ipmiconfig
```

Usage Notes

- This command attempts to locate and access the IPMI configuration stored in the Oracle Cluster Registry (OCR) and should be executed under the account used to install Oracle Clusterware, or an authorization failure may be reported.
- An authorization failure may not result when executed under another account, if the registry contains no IPMI configuration data.

Example

The `crsctl query css ipmiconfig` command returns output similar to the following:

```
CRS-4236: Oracle Clusterware configured to use IPMI
```

Or

```
CRS-4237: Oracle Clusterware is not fully configured to use IPMI
```

crsctl query css ipmidevice

Use the `crsctl query css ipmiconfig` command to determine the presence of the Intelligent Platform Management Interface (IPMI) driver on the local system.

Syntax

```
crsctl query css ipmidevice
```

Usage Notes

- This command performs a pre-check during IPMI installation, and is normally issued only by the installer.
- This command performs a perfunctory check and a success return does not guarantee that the IPMI hardware is fully configured for use.
- There are no special privileges required to run this command.

Example

The `crsctl query css ipmidevice` command returns output similar to the following:

```
CRS-4231: IPMI device and/or driver found
```

Or

```
CRS-4218: Unable to access an IPMI device on this system
```

crsctl query css votedisk

Use the `crsctl query css votedisk` command to display the voting files used by Cluster Synchronization Services, the status of the voting files, and the location of the disks, whether they are stored on Oracle ASM or elsewhere.

Syntax

```
crsctl query css votedisk
```

Example

The `crsctl query css votedisk` command returns output similar to the following:

```
$ crsctl query css votedisk
##  STATE    File Universal Id                File Name Disk group
```

```
--  -----    -----------------                 --------- ---------
1. ONLINE   296641fd201f4f3fbf3452156d3b5881 (/ocfs2/host09_vd3) []
2. ONLINE   8c4a552bdd9a4fd9bf93e444223146f2 (/netapp/ocrvf/newvd) []
3. ONLINE   8afeee6ae3ed4fe6bfbb556996ca4da5 (/ocfs2/host09_vd1) []
Located 3 voting file(s).
```

crsctl query dns

Use the `crsctl query dns` command to obtain a list of addresses returned by DNS lookup of the name with the specified DNS server.

Syntax

```
crsctl query dns {-servers | -name name [-dnsserver DNS_server_address]
[-port port] [-attempts number_of_attempts] [-timeout timeout_in_seconds] [-v]}
```

Parameters

Table E–50 crsctl query dns Command Parameters

Parameter	Description
`-servers`	Use the `-servers` parameter to list the current DNS configuration of the node on which you run the command. Typically, on Linux/UNIX, CRSCTL reads the `/etc/resolv.conf` file at start time and that is what the system configuration is until you restart or until the resolver is restarted. CRSCTL gets its information from resolver. You can use this parameter or, optionally, you can specify the other parameters in the command syntax string.
`-name` *name*	Specify the fully-qualified domain name you want to look up.
`-dnsserver` *DNS_server_address*	Specify the address of the DNS server on which you want the domain name to be looked up.
`-port` *port*	The port on which the DNS server listens. If you do not specify a port, then it defaults to port 53.
`-attempts` *number_of_attempts*	Specify the number of retry attempts.
`-timeout` *timeout_in_seconds*	Specify the timeout length in seconds.
`-v`	Verbose output.

Example

The `crsctl query dns` command returns output similar to the following for a DNS server named `stmjk07-vip.stmjk0462.foo.com`:

```
CRS-10024: DNS server returned 192.168.29.250 for name
stmjk07-vip.stmjk0462.foo.com
```

If you choose the `-servers` parameter, then the command returns output similar to the following:

```
CRS-10018: the following configuration was found on the system:
CRS-10019: There are 3 domains in search order. They are:
us.foo.com
foo.com
foocorp.com
CRS-10022: There are 3 name servers. They are:
```

```
192.168.249.41
192.168.249.52
192.168.202.15
CRS-10020: number of retry attempts for name lookup is: 2
CRS-10021: timeout for each name lookup is: 1
```

crsctl query socket udp

Use the `crsctl query socket udp` command to verify that a daemon can listen on specified address and port.

Syntax

```
crsctl query socket udp [-address address] [-port port]
```

Table E–51 crsctl query socket udp

Parameter	Description
`-address` *address*	Specify the IP address on which the socket is to be created. If you do not specify an address, then CRSCTL assumes the local host as the default.
`-port` *port*	Specify the port on which the socket is to be created. If you do not specify a port, then CRSCTL assumes 53 as the default.

Usage Notes

- You must run this command as `root` to verify port numbers less than 1024.

Examples

The following examples show various outputs:

```
$ crsctl query socket udp
CRS-10030: could not verify if port 53 on local node is in use

# crsctl query socket udp
CRS-10026: successfully created socket on port 53 on local node
```

The first of the preceding two commands was not run as `root`, and in both commands no port was specified, so CRSCTL assumed the default, 53, which is less than 1024. This condition necessitates running the command as `root`.

```
$ crsctl query socket udp -port 1023
CRS-10030: could not verify if port 1023 on local node is in use

# crsctl query socket udp -port 1023
CRS-10026: successfully created socket on port 1023 on local node
```

Similar to the first two examples, the first of the preceding two commands was not run as `root`, and, although a port number was specified, it is still less than 1024, which requires `root` privileges to run the command.

In this last example, a port number greater than 1024 is specified, so there is no need to run the command as `root`:

```
$ crsctl query socket udp -port 1028
CRS-10026: successfully created socket on port 1028 on local node
```

crsctl release dhcp

Use the `crsctl release dhcp` command to send a DHCP lease release request to a specific client ID and send release packets on the network to a specific port.

Syntax

```
crsctl release dhcp -clientid clientid [-port port]
```

Parameters

Table E–52 crsctl release dhcp Command Parameters

Parameter	Description
`-clientid clientid`	Specify the client ID for which you want to attempt release. Obtain the client ID by running the `crsctl get clientid` command.
`-port port`	The port to which CRSCTL sends the release packets. If you do not specify a port, then CRSCTL uses the default value 67.

Example

The `crsctl release dhcp` command returns output similar to the following:

```
$ crsctl release dhcp -clientid  stmjk0462clr-stmjk01-vip

CRS-10012: released DHCP server lease for client ID stmjk0462clr-stmjk01-vip
on port 67
```

crsctl relocate resource

Use the `crsctl relocate resource` command to relocate resources to another server in the cluster.

Syntax

```
crsctl relocate resource {resource_name [-k cid] | {resource_name | -all}
-s source_server | -w "filter"} [-n destination_server] [-env
"env1=val1,env2=val2,..."]
[-i] [-f]
```

Parameters

Table E–53 crsctl relocate resource Command Parameters

Parameter	Description
`resource_name [-k cid]`	The name of a resource you want to relocate. Optionally, you can also specify the resource cardinality ID. If you specify this parameter, then Oracle Clusterware relocates the resource instance that have the cardinality you specify.

`resource_name | -all`
`-s source_server`
	Specify one particular or all resources located on a particular server *from* which you want to relocate those resources.

Table E–53 (Cont.) crsctl relocate resource Command Parameters

Parameter	Description
`-w "`*`filter`*`"`	Specify a resource filter that Oracle Clusterware uses to limit the number of resources relocated. The filter must be enclosed in double quotation marks (`""`). Examples of resource filters include: ■ `"TYPE == cluster_resource"`: This filter limits Oracle Clusterware to relocate only resources of `cluster_resource` type ■ `"CHECK_INTERVAL > 10"`: This filter limits Oracle Clusterware to relocate resources that have a value greater than 10 for the `CHECK_INTERVAL` resource attribute ■ `"(CHECK_INTERVAL > 10) AND (NAME co 2)"`: This filter limits Oracle Clusterware to relocate resources that have a value greater than 10 for the `CHECK_INTERVAL` resource attribute *and* the name of the resource contains the number `2` **See Also:** "Filters" on page E-2 for more information
`-n` *`destination_server`*	Specify the name of the server *to* which you want relocate resources. If you do not specify a destination server, then Oracle Clusterware relocates the resources to the best server according to the attribute profile of each resource.
`-env "`*`env1=val1, env2=val2`*`,..."`	You can optionally override one or more resource profile attribute values for this command. If you specify multiple environment name-value pairs, then you must separate each pair with a comma and enclose the entire list in double quotation marks (`""`).
`-i`	If you specify `-i`, then the command returns an error if processing this command requires waiting for Oracle Clusterware to unlock the resource or its dependents. Sometimes, Oracle Clusterware locks resources or other objects to prevent commands from interfering with each other.
`-f`	Specify the `-f` parameter to force the relocating of the resource when it has other resources running that depend on it. Dependent resources are relocated or stopped when you use this parameter. **Note:** When you are relocating resources that have cardinality greater than 1, you must use either `-k` or `-s` to narrow down which resource instances are to be relocated.

Usage Notes

- Any one of the three following options is required to specify which resources you want to relocate:
  - You can specify one particular resource to relocate.
  - Or you can specify one particular or all the resources to relocate from a particular source server.
  - Thirdly, you can specify a resource filter that Oracle Clusterware uses to match resources to relocate.
- If a resource has a degree ID greater than 1, then Oracle Clusterware relocates all instances of the resource.

- You must have read and execute permissions on the specified resources to relocate them
- Do not use this command for any resources with names that begin with *ora* because these resources are Oracle resources.

Example

To relocate one particular resource from one server to another:

```
# crsctl relocate resource myResource1 -s node1 -n node3
```

crsctl relocate server

Use the `crsctl relocate server` command to relocate a server to a different server pool.

Syntax

```
crsctl relocate server server_name [...] -c server_pool_name [-i] [-f]
```

Parameters

Table E–54 crsctl relocate server Command Parameters

Parameter	Description
server_name	The name of the server you want to relocate. You can provide a space-delimited list of servers to relocate multiple servers.
-c *server_pool_name*	Specify the name of the server pool *to* which you want relocate the servers.
-i	If you specify `-i`, then the command fails if Oracle Clusterware cannot process the request immediately.
-f	If you specify the `-f` parameter, then Oracle Clusterware stops resources running on the servers in another server pool and relocates that server into the server pool you specified. If you do not specify the `-f` parameter, then Oracle Clusterware checks for resources that must be stopped on the servers that are being relocated. If it finds any, then Oracle Clusterware rejects the `crsctl relocate server` command. **Note:** If the number of servers in the server pool is not above the value of the `MIN_SIZE` server pool attribute, then the force parameter will have no affect because CRSCTL will not violate the configuration.

Usage Notes

- The *server_name* and `-c` *server_pool_name* parameters are required

Example

To move the `node6` and `node7` servers into the `sp1` server pool without disrupting any active resources on those nodes, use the following command:

```
$ crsctl relocate server node6 node7 -c sp1
```

crsctl replace discoverystring

Use the `crsctl replace discoverystring` command to replace the existing discovery string used to locate voting files.

Syntax

```
crsctl replace discoverystring "absolute_path[,...]"
```

Parameters

Table E–55 crsctl replace discoverystring Command Parameters

Parameter	Description
absolute_path	Specify one or more comma-delimited absolute paths that match one or more voting file locations. Wildcards may be used. The list of paths must be enclosed in double quotation marks ("").

Usage Notes

- You must be `root`, the Oracle Clusterware installation owner, or a member of the Administrators group to run this command.
- You can run this command on any node in the cluster.
- If you store voting files in an Oracle ASM disk group, then you cannot change the discovery string.

Example

Assume the current discovery string is `/oracle/css1/*`. To also use voting files in the `/oracle/css2/` directory, replace the current discovery string using the following command:

```
# crsctl replace discoverystring "/oracle/css1/*,/oracle/css2/*"
```

crsctl replace votedisk

Use the `crsctl replace votedisk` command to move or replace the existing voting files. This command creates voting files in the specified locations, either in Oracle ASM or some other storage option. Oracle Clusterware copies existing voting file information into the new locations and removes the voting files from the former locations.

Syntax

```
crsctl replace votedisk [+asm_disk_group | path_to_voting_disk [...]]
```

Parameters

Table E–56 crsctl replace votedisk Command Parameters

Parameter	Description
+*asm_disk_group*	Specify the disk group in Oracle ASM where you want to locate the voting file.
path_to_voting_disk [...]	A space-delimited list of voting file paths for voting files that reside outside of Oracle ASM.

Usage Notes

- You must be `root`, the Oracle Clusterware installation owner, or a member of the Administrators group to run this command.
- Specify to replace a voting file in either an Oracle ASM disk group or in some other storage device.
- You can run this command on any node in the cluster.

Example

Example 1

To replace a voting file that is located within Oracle ASM:

```
$ crsctl replace votedisk +diskgroup1
```

Example 2

To replace a voting file that is located on a shared file system:

```
$ crsctl replace votedisk /mnt/nfs/disk1 /mnt/nfs/disk2
```

crsctl request action

Use the `crsctl request action` command to perform a specific action on specific resource.

Syntax

```
crsctl request action action_name {-r resource_name [...] | -w "filter"} [-env
"env1=val1,env2=val2,..."] [-i]
```

Parameters

Table E–57 crsctl request action Command Parameters

Parameter	Description
action_name	Specify the name of the action you want perform. Actions supported by a particular resource are listed in the `ACTIONS` resource attribute of that resource.
`-r` *resource_name* `[...]`	Specify a particular resource. Multiple resource names must be separated by a space.

Table E–57 (Cont.) crsctl request action Command Parameters

Parameter	Description
`-w "`*`filter`*`"`	Alternative to specifying resource names, you can specify a resource filter that Oracle Clusterware uses to limit the number of resources on which actions are performed. Examples of resource filters include: ■ `TYPE == cluster_resource`: This filter limits Oracle Clusterware to perform actions on only resources of `cluster_resource` type ■ `CHECK_INTERVAL > 10`: This filter limits Oracle Clusterware to perform actions on only resources that have a value greater than 10 for the `CHECK_INTERVAL` resource attribute ■ `(CHECK_INTERVAL > 10) AND (NAME co 2)`: This filter limits Oracle Clusterware to perform actions on only resources that have a value greater than 10 for the `CHECK_INTERVAL` resource attribute *and* the name of the resource contains the number `2` **See Also:** "Filters" on page E-2 for more information
`-env "`*`env1=val1, env2=val2,...`*`"`	You can optionally override one or more resource profile attribute values with the `-env` command parameter. If you specify multiple environment name-value pairs, then you must separate each pair with a comma and enclose the entire list in double quotation marks (`""`).
`-i`	If you specify `-i`, then the command fails if Oracle Clusterware cannot process the request immediately.

Usage Notes

The *`action_name`* must be a supported action, defined both in the agent and the `ACTIONS` resource attribute and the user must have permission to run the action. You can set permissions in the `ACTIONS` resource attribute.

Example

To initiate the specified action inside the application specific agent of a particular resource:

```
$ crsctl request action action1 -r res1
```

crsctl request dhcp

Use the `crsctl request dhcp` command to send DHCP request packets on the network at the specified port. If the DHCP server has an IP address it can provide, then it responds with the IP address for the client ID.

Syntax

```
crsctl request dhcp -clientid clientid [-port port]
```

Parameters

Table E–58 crsctl request dhcp Command Parameters

Parameter	Description
`-clientid` *`clientid`*	Specify the client ID for which you want to attempt request. Obtain the client ID by running the `crsctl get clientid` command.
`-port` *`port`*	The port to which CRSCTL sends the request packets. If you do not specify a port, then CRSCTL uses the default value 67.

Example

The `crsctl request dhcp` command returns output similar to the following:

```
$ crsctl request dhcp -clientid stmjk0462clr-stmjk01-vip

CRS-10009: DHCP server returned server: 192.168.53.232,
 loan address : 192.168.29.228/255.255.252.0, lease time: 43200
```

crsctl set cluster hubsize

Use the `crsctl set cluster hubsize` command to set the maximum number of Hub Nodes for an Oracle Flex Cluster.

Syntax

```
crsctl set cluster hubsize
```

Example

The following command example sets the maximum number of Hub Nodes to 32:

```
$ crsctl set cluster hubsize 32
```

crsctl set cluster mode

Use the `crsctl set cluster mode` command to change a cluster to an Oracle Clusterware standard Cluster or an Oracle Flex Cluster.

Syntax

```
crsctl set cluster mode [standard | flex]
```

Usage Notes

- Choose either `standard` or `flex` depending on how you want to configure the cluster.

crsctl set cpu equivalency

Use the `crsctl set cpu equivalency` command to set a value for the `CPU_EQUIVALENCY` server configuration attribute.

Syntax

```
crsctl set cpu equivalency
```

crsctl set crs autostart

Use the `crsctl set crs autostart` command to set the Oracle Clusterware automatic resource start criteria. The autostart delay and minimum server count criteria delay Oracle Clusterware resource autostart until one of the two conditions are met.

Syntax

```
crsctl set crs autostart [delay delay_time] [servercount count]
```

Table E–59 crsctl set crs autostart Command Parameters

Parameter	Description
`delay` *`delay_time`*	Specify the number of seconds to delay Oracle Clusterware autostart.
`servercount` *`count`*	Specify the minimum number of servers required for Oracle Clusterware autostart.

Example

To ensure that Oracle Clusterware delays resource autostart for 60 seconds after the first server in the cluster is ONLINE:

```
crsctl set crs autostart delay 60
```

To ensure that Oracle Clusterware waits for there to be at least two servers ONLINE before it initiates resource autostart:

```
crsctl set crs autostart servercount 2
```

To ensure that Oracle Clusterware delays resource autostart until either of the previous two conditions are met (in no particular order):

```
crsctl set crs autostart delay 60 servercount 2
```

crsctl set css

Use the `crsctl set css` command to set the value of a Cluster Synchronization Services parameter.

Syntax

```
crsctl set css parameter value
```

Usage Notes

- Do not use the `crsctl set css` command to set the following parameters unless instructed to do so by My Oracle Support.
- Cluster Synchronization Services parameters include:

  ```
  diagwait
  disktimeout
  logfilesize
  misscount
  priority
  reboottime
  ```

crsctl set css ipmiaddr

Use the `crsctl set css ipmiaddr` command to store the address of the local Intelligent Platform Management Interface (IPMI) device in the Oracle Local Registry.

Syntax

```
crsctl set css ipmiaddr ip_address
```

Usage Notes

- Run the command under the user account used to install Oracle Clusterware
- Obtain the IP address used by the IPMI device using either `ipmiutil` or `ipmitool` as `root` on the local server
- Oracle Clusterware stores the IP address for IPMI in the configuration store, and distributes the address as required
- This command only stores the IPMI IP address on the server from which you run it
- This command fails if another server cannot access IPMI at the supplied address

Example

To store the IPMI IP address on a local server and distribute it to other cluster nodes:

```
$ crsctl set css ipmiaddr 192.0.2.244
```

crsctl set css ipmiadmin

Use the `crsctl set css ipmiadmin` command to store the login credentials of an Intelligent Platform Management Interface (IPMI) administrator in the Oracle Local Registry.

Syntax

```
crsctl set css ipmiadmin ipmi_administrator_name
```

Usage Notes

- This command must be run under the user account that installed Oracle Clusterware.
- When prompted, provide the new password to associate with the new administrator account name. Oracle Clusterware stores the name and password for the local IPMI in the configuration store, and distributes the new credentials as required.
- This command only modifies the IPMI administrator on the server from which you run it.
- This command fails if another server cannot access the local IPMI at the supplied address.

Example

To modify the IPMI administrator `scott`:

```
$ crsctl set css ipmiadmin scott
```

crsctl set css leafmisscount

Use the `crsctl set css leafmisscount` command to specify, in seconds, the amount of time that must pass without any communication between a Leaf Node and the Hub Node to which it is attached, before the connection is declared to be no longer active and the Leaf Node is removed from the cluster.

Syntax

```
crsctl set css leafmisscount number_of_seconds
```

Usage Notes

- You must run this command as `root` or the Oracle Clusterware installation owner
- You can only run this command on a Hub Node

Example

To configure a 30 second interval between communication failure and removal of the Leaf Node from the cluster:

```
$ crsctl set css leafmisscount 30
```

crsctl set node role

Use the `crsctl set node role` command to set the role of a specific node in the cluster.

Syntax

```
crsctl set node role [-node node_name] {hub | leaf}
```

Usage Notes

- You can specify a particular node for which to set role information. If you do not specify a particular node, then CRSCTL sets the node role on the local node.
- Specify the `hub` option to configure the node role as a Hub Node.
- Specify the `leaf` option to configure the node role as a Leaf Node.
- You must restart the Oracle Clusterware technology stack to apply a node role change.

Example

To configure a node as a Hub Node:

```
$ crsctl set node role -node node151 hub
```

crsctl set resource use

Use the `crsctl set resource use` command to set the value of the `RESOURCE_USE_ENABLED` server configuration parameter for the server on which you run this command.

Syntax

```
crsctl set resource use [1 | 0]
```

Usage Notes

- The possible values are 1 or 0. If you set the value for this attribute to 1, which is the default, then the server can be used for resource placement. If you set the value to 0, then Oracle Clusterware disallows starting server pool resources on the server. The server remains in the Free server pool.
- You must run this command as `root` or a cluster administrator, or an administrator on Windows systems.

Example

To set the value of the `RESOURCE_USE_ENABLED` server configuration parameter:

```
# crsctl set resource use 1
```

crsctl set server label

Use the `crsctl set server label` command to set the configuration value of the `SERVER_LABEL` server configuration attribute for the server on which you run this command.

Syntax

```
crsctl set server label value
```

Usage Notes

- Specify a value for the server. This value can reflect a physical location, such as `building_A`, or some other identifying characteristic of the server, such as `hubserver`.
- You must restart the Oracle Clusterware technology stack on the node before any changes you make take effect.

Example

The `crsctl set server label` command returns output similar to the following:

```
$ crsctl set server label hubserver
```

crsctl setperm serverpool

Use the `crsctl setperm serverpool` command to set permissions for a particular server pool.

Syntax

```
crsctl setperm serverpool server_pool_name {-u acl_string | -x acl_string |
-o user_name | -g group_name}
```

Parameters

Table E–60 crsctl setperm serverpool Command Parameters

Parameter	Description
server_pool_name	Specify the name of the server pool for which you want to set permissions.

Table E–60 (Cont.) crsctl setperm serverpool Command Parameters

Parameter	Description

`{-u | -x | -o | -g}`	You can specify only one of the following parameters for a server pool:
	- `-u` *`acl_string`*: You can update the access control list (ACL) for a server pool
	- `-x` *`acl_string`*: You can delete the ACL for a server pool
	- `-o` *`user_name`*: You can change the owner of a server pool by entering a user name
	- `-g` *`group_name`*: You can change the primary group of a server pool by entering a group name
	Specify a user, group, or other ACL string, as follows:
	`user:`*`user_name`*`[:`*`readPermwritePermexecPerm`*`] |`
	`group:`*`group_name`*`[:`*`readPermwritePermexecPerm`*`] |`
	`other[::`*`readPermwritePermexecPerm`*`]`
	- `user`: User ACL
	- `group`: Group ACL
	- `other`: Other ACL
	- `readPerm`: Read permission for the server pool; the letter `r` grants a user, group, or other read permission, the minus sign (`-`) denies read permission
	- `writePerm`: Write permission for the server pool; the letter `w` grants a user, group, or other write permission, the minus sign (`-`) denies write permission
	- `execPerm`: Execute permission for the server pool; the letter `x` grants a user, group, or other execute permission, the minus sign (`-`) denies execute permission

Usage Notes

- The *`server_pool_name`* parameter is required
- Do not use this command for any server pools with names that begin with *ora* because these server pools are Oracle server pools
- While you can use this command in either environment, it is only useful in the Oracle RAC environment

Example

To grant read, write, and execute permissions on a server pool for user Jane Doe:

```
crsctl setperm serverpool sp3 -u user:jane.doe:rwx
```

crsctl start cluster

Use the `crsctl start cluster` command on any node in the cluster to start the Oracle Clusterware stack.

Syntax

```
crsctl start cluster [-all | -n server_name [...]]
```

Usage Notes

- You can choose to start the Oracle Clusterware stack on all servers in the cluster, on one or more named servers in the cluster (separate multiple server names by a space), or the local server, if you do not specify either `-all` or `-n`.
- You can use this cluster-aware command on any node in the cluster.

Example

To start the Oracle Clusterware stack on two named servers run the following command as `root`:

```
# crsctl start cluster -n node1 node2
```

crsctl start crs

Use the `crsctl start crs` command to start Oracle High Availability Services on the local server.

Syntax

```
crsctl start crs [-excl [-nocrs] [-cssonly]] | [-wait | -waithas | -nowait] |
[-noautostart]
```

Parameters

Table E–61 crsctl start crs Command Parameters

Parameter	Description
`-excl`	Starts Oracle Clusterware in exclusive mode with two options: ■ Specify the `-nocrs` parameter to start Oracle Clusterware in exclusive mode without starting CRSD. ■ Specify the `-cssonly` parameter to start CSSD, only.
`-wait | -waithas | -nowait`	Choose one of the following: ■ Specify `-wait` to wait until startup is complete and display all progress and status messages. ■ Specify `-waithas` to wait until startup is complete and display OHASD progress and status messages. ■ Specify `-nowait` to not wait for OHASD to start
`-noautostart`	Start only OHASD.

Usage Notes

- You must run this command as `root`
- This command starts Oracle High Availability Services only on the local server

Example

To start Oracle High Availability Services on the local server, run the following command as `root`:

```
# crsctl start crs
```

crsctl start ip

Use the `crsctl start ip` command to start a given IP name or IP address on a specified interface with a specified subnet mask. Run this command on the server on which you want to start the IP.

Syntax

```
crsctl start ip -A {IP_name | IP_address}/netmask/interface_name
```

Parameters

Table E–62 crsctl start ip Command Parameters

Parameter	Description
`{IP_name | IP_address}`	Specify either a domain name or an IP address. If you do not specify a fully-qualified domain name, then CRSCTL uses a standard name search.
netmask	Specify a subnet mask for the IP to start.
interface_name	Specify an interface on which to start the IP.

Example

To start an IP on the local server, run the command similar to the following:

```
$ crsctl start ip -A 192.168.29.220/255.255.252.0/eth0
```

crsctl start rollingpatch

The `crsctl start rollingpatch` command transitions Oracle Clusterware and Oracle ASM into rolling patch mode. In this mode, the software tolerates nodes having different patch levels.

Syntax

```
crsctl start rollingpatch
```

Usage Notes

- This command queries the Oracle Clusterware rolling patch state and Oracle ASM cluster state. If either one is not in rolling patch mode, it will use the appropriate method to transition Oracle Clusterware or Oracle ASM to rolling patch mode.
- If Oracle Clusterware and Oracle ASM are both in rolling patch mode when you run this command, then this command does nothing.
- The rolling patch mode is not persistent. If all the nodes in a cluster are shut down and restarted, then the cluster transitions out of rolling patch mode when it is restarted. Similarly, if Oracle Clusterware is stopped and then restarted on all nodes in the cluster, then the rolling patch mode is lost.
- This command does not transition Oracle ASM to rolling patch mode if issued within an Oracle ASM Client Cluster.

crsctl start rollingupgrade

The `crsctl start rollingupgrade` command transitions Oracle Clusterware and Oracle ASM into rolling upgrade mode.

Syntax

```
crsctl start rollingupgrade version
```

Usage Notes

- This command queries the Oracle Clusterware rolling upgrade state and Oracle ASM cluster state. If either one is not in rolling upgrade mode, it will use the appropriate method to transition Oracle Clusterware or Oracle ASM to rolling upgrade mode.
- If Oracle Clusterware and Oracle ASM are both in rolling upgrade mode when you run this command, then this command does nothing.
- The rolling upgrade mode is not persistent. If all the nodes in a cluster are shut down and restarted, then the cluster transitions out of rolling upgrade mode when it is restarted. Similarly, if Oracle Clusterware is stopped and then restarted on all nodes in the cluster, then the rolling upgrade mode is lost.
- This command does not transition Oracle ASM to rolling upgrade mode if run within an Oracle ASM Client Cluster.

crsctl start testdns

Use the `crsctl start testdns` command to start a test DNS server that will listen on a specified IP address and port. The test DNS server does not respond to incoming packets but does display the packets it receives. Typically, use this command to check if domain forwarding is set up correctly for the GNS domain.

Syntax

```
crsctl start testdns [-address address [-port port]] [-once] [-v]
```

Parameters

Table E–63 crsctl start testdns Command Parameters

Parameter	Description
`-address` *`address`*	Specify a server address in the form *`IP_address`*`/`*`netmask`* `[/`*`interface_name`*`]`.
`-port` *`port`*	The port on which the server listens. If you do not specify a port, then it defaults to port 53.
`-once`	Specify this flag to indicate that the DNS server should exit after it receives one DNS query packet.
`-v`	Verbose output.

Example

To start a test DNS server on the local server, run the command similar to the following:

```
$ crsctl start testdns -address 192.168.29.218 -port 63 -v
```

crsctl status category

Use the `crsctl status category` command to obtain information about a server category.

Syntax

```
crsctl status category {category_name [category_name [...]] | [-w "filter" |
    -server server_name]}
```

Parameters

Table E–64 crsctl status category Command Parameters

Parameter	Description
category_name	Specify the name of the server category or a space-delimited list of server categories for which you want to obtain the status.
-w "*filter*"	Alternatively, you can specify a category filter preceded by the -w flag. **See Also:** "Filters" on page E-2 for more information
-server *server_name*	Alternatively, you can specify a particular server to list all of the categories that the server matches.

Examples

To obtain the status of a server category using filters:

```
$ crsctl stat category -w "ACTIVE_CSS_ROLE = hub"

NAME=my_category_i
ACL=owner:mjkeenan:rwx,pgrp:svrtech:rwx,other::r--
ACTIVE_CSS_ROLE = hub
EXPRESSION=(CPU_COUNT > 3)
```

To obtain the status of a server category by server:

```
$ crsctl stat category -server node1

NAME=my_category
ACL=owner:mjkeenan:rwx,pgrp:svrtech:rwx,other::r--
ACTIVE_CSS_ROLE = hub
EXPRESSION=
```

crsctl status ip

Use the `crsctl status ip` command to check if a given IP address is up on the network.

Syntax

```
crsctl status ip -A {IP_name | IP_address}
```

Parameters

Table E–65 crsctl status ip Command Parameters

Parameter	Description
{*IP_name* | *IP_address*}	Specify either a domain name or an IP address. If you do not specify a fully-qualified domain name, then CRSCTL uses a standard name search.

Example

The `crsctl status ip` command returns output similar to the following:

```
CRS-10003: IP address 192.168.29.220 could be reached from current node
```

crsctl status policy

Use the `crsctl status policy` command to view the status and definition of a configuration policy.

Syntax

```
crsctl status policy [policy_name [policy_name [...]] | -w "filter" | -active]
```

Parameters

Table E–66 crsctl status policy Command Parameters

Parameter	Description
policy_name	Specify the name of the policy or a space-delimited list of policy names for which you want to view the status.
-w "*filter*"	Alternatively, you can specify a policy filter preceded by the -w flag. **See Also:** "Filters" on page E-2 for more information
-active	Alternatively, you can specify this parameter to display the status of the active policy.

Usage Notes

- Privileges necessary to run this command depend on the value of the `ACL` attribute of the policy set

crsctl status policyset

Use the `crsctl status policyset` command to view the current policies in the policy set, including the access control list, which governs who can modify the set, the last activated policy, and the configuration which is now in effect, which is known as the Current policy.

Syntax

```
crsctl status policyset [-file file_name]
```

Parameters

Table E–67 crsctl status policyset Command Parameters

Parameter	Description
-file *file_name*	You can specify this parameter to create a file that you can edit and then send back using `crsctl modify policyset` to add, delete, or update multiple policies. If you do not specify this optional parameter, then CRSCTL displays the Current configuration.

Usage Notes

- Privileges necessary to run this command depend on the value of the ACL attribute of the policy set

Example

This command returns output similar to the following:

```
ACL=owner:'mjkeenan:rwx,pgrp:g900:rwx,other::r--'
LAST_ACTIVATED_POLICY=DayTime
SERVER_POOL_NAMES=Free pool1 pool2 pool3
POLICY
NAME=DayTime
DESCRIPTION=Test policy
SERVERPOOL
  NAME=pool1
  IMPORTANCE=0
  MAX_SIZE=2
  MIN_SIZE=2
  SERVER_CATEGORY=
  SERVER_NAMES=
SERVERPOOL
  NAME=pool2
  IMPORTANCE=0
  MAX_SIZE=1
  MIN_SIZE=1
  SERVER_CATEGORY=
SERVERPOOL
  NAME=pool3
  IMPORTANCE=0
  MAX_SIZE=1
  MIN_SIZE=1
  SERVER_CATEGORY=
POLICY
NAME=NightTime
DESCRIPTION=Test policy
SERVERPOOL
  NAME=pool1
  IMPORTANCE=0
  MAX_SIZE=1
  MIN_SIZE=1
  SERVER_CATEGORY=
SERVERPOOL
  NAME=pool2
  IMPORTANCE=0
  MAX_SIZE=2
  MIN_SIZE=2
  SERVER_CATEGORY=
SERVERPOOL
  NAME=pool3
  IMPORTANCE=0
  MAX_SIZE=1
  MIN_SIZE=1
  SERVER_CATEGORY=
POLICY
NAME=Weekend
DESCRIPTION=Test policy
SERVERPOOL
  NAME=pool1
  IMPORTANCE=0
  MAX_SIZE=0
  MIN_SIZE=0
  SERVER_CATEGORY=
SERVERPOOL
  NAME=pool2
  IMPORTANCE=0
  MAX_SIZE=1
  MIN_SIZE=1
  SERVER_CATEGORY=
SERVERPOOL
  NAME=pool3
  IMPORTANCE=0
  MAX_SIZE=3
  MIN_SIZE=3
  SERVER_CATEGORY=
```

crsctl status server

Use the `crsctl status server` command to obtain the status and configuration information of one or more particular servers.

Syntax

```
crsctl status server {server_name [...] | -w "filter"} [-g | -p | -v | -f] |
    [-category category_name | -w "filter"]
```

Parameters

Table E–68 crsctl status server Command Parameters

Parameter	Description
server_name [...]	Specify one or more space-delimited server names.
-w "*filter*"	Specify a filter to determine which servers are displayed. The filter must be enclosed in double quotation marks (""). Values that contain parentheses or spaces must be enclosed in single quotation marks (''). For example, "STATE = ONLINE" limits the display to servers that are online. **See Also:** "Filters" on page E-2 for more information
-g | -p | -v | -f	You can specify one of the following parameters when Oracle Clusterware checks the status of specific servers: ■ -g: Use this parameter to check if the specified servers are registered ■ -p: Use this parameter to display static configuration of the specified servers ■ -v: Use this parameter to display the run-time configuration of the specified servers ■ -f: Use this parameter to display the full configuration of the specified servers
-category *category_name*	You can specify a particular category of servers for which to obtain status.
-w "*filter*"	Specify a filter to determine which categories are displayed. The filter must be enclosed in double quotation marks (""). Values that contain parentheses or spaces must be enclosed in single quotation marks (''). For example, "STATE = ONLINE" limits the display to servers that are online. **See Also:** "Filters" on page E-2 for more information

Example

Example 1

The `crsctl status server` command returns output similar to the following:

```
NAME=node1
STATE=ONLINE

NAME=node2
STATE=ONLINE
```

Example 2

The full configuration of a specific server is similar to the following:

```
NAME=node2
MEMORY_SIZE=72626
CPU_COUNT=12
CPU_CLOCK_RATE=1711
CPU_HYPERTHREADING=0
CPU_EQUIVALENCY=1000
DEPLOYMENT=other
CONFIGURED_CSS_ROLE=hub
RESOURCE_USE_ENABLED=1
SERVER_LABEL=
PHYSICAL_HOSTNAME=
STATE=ONLINE
ACTIVE_POOLS=ora.pool1
STATE_DETAILS=
ACTIVE_CSS_ROLE=hub
```

crsctl status serverpool

Use the `crsctl status serverpool` command to obtain the status and configuration information of one or more particular server pools.

Syntax

```
crsctl status serverpool [server_pool_name [...] | -w "filter"] [-p | -v | -f]

crsctl status serverpool {[server_pool_name [...]} -g
```

Parameters

Table E–69 crsctl status serverpool Command Parameters

Parameter	Description
`[`*server_pool_name*` [...] -g`	Specify one or more space-delimited server pool names to identify specific server pools. `-g`: Use this parameter to check if the specified server pools are registered **Note:** You *cannot* use the `-g` parameter with any other parameters after you specify the server pool names.
`[-w "`*filter*`"]`	Use this parameter to specify a filter, such as `MIN_SIZE > 3`, surrounded by double quotation marks (`""`). Use this parameter to identify server pools by a particular characteristic. **See Also:** "Filters" on page E-2 for more information

Table E–69 (Cont.) crsctl status serverpool Command Parameters

Parameter	Description
`[-p | -v | -f]`	You can optionally specify one of the following parameters: ■ `-p`: Use this parameter to display static configuration of the specified server pools ■ `-v`: Use this parameter to display the run-time configuration of the specified server pools ■ `-f`: Use this parameter to display the full configuration of the specified server pools

Usage Notes

- The *`server_pool_name`* parameter or a filter is required
- Do not use this command for any server pools with names that begin with *ora* because these server pools are Oracle server pools
- While you can use this command in either environment, it is only useful in the Oracle RAC environment

Examples

Example 1

To display the full configuration of the server pool `sp1`:

```
$ crsctl status serverpool sp1 -f
NAME=spl
IMPORTANCE=1
MIN_SIZE=0
MAX_SIZE=-1
SERVER_NAMES=node3 node4 node5
PARENT_POOLS=Generic
EXCLUSIVE_POOLS=
ACL=owner:oracle:rwx,pgrp:oinstall:rwx,other::r--
SERVER_CATEGORY=ora.hub.category
ACTIVE_SERVERS=node3 node4
```

Example 2

To display all the server pools and the servers associated with them, use the following command:

```
$ crsctl status serverpool
NAME=Free
ACTIVE_SERVERS=

NAME=Generic
ACTIVE_SERVERS=node1 node2

NAME=ora.orcl
ACTIVE_SERVERS=node1 node2

NAME=sp1
ACTIVE_SERVERS=node3 node4
```

Example 3

To find a server pool that meets certain criteria, use the following command:

```
$ crsctl status serverpool -w "MAX_SIZE > 1"
NAME=sp2
ACTIVE_SERVERS=node3 node4
```

crsctl status testdns

Use the `crsctl status testdns` command to query the test DNS server running on a specified address and local host name.

Syntax

```
crsctl status testdns [-server DNS_server_address] [-port port] [-v]
```

Parameters

Table E–70 crsctl status testdns Command Parameters

Parameter	Description
`-server` *DNS_server_address*	Specify the DNS server address for which you want to check the status.
`-port` *port*	The port on which the DNS server listens. If you do not specify a port, then it defaults to port 53.
`-v`	Verbose output.

Example

The `crsctl status testdns` command returns output similar to the following:

```
CRS-10024: DNS server returned 192.168.28.74 for name
stmjk07-vip.stmjk0462.foo.com
```

crsctl stop cluster

Use the `crsctl stop cluster` command on any node in the cluster to stop the Oracle Clusterware stack on all servers in the cluster or specific servers.

Syntax

```
crsctl stop cluster [-all | -n server_name [...]] [-f]
```

Usage Notes

- If you do not specify `-all` or one or more space-delimited server names, then Oracle Clusterware stops the Oracle Clusterware stack on the local server.
- You can use this cluster-aware command on any node in the cluster.
- This command attempts to gracefully stop resources managed by Oracle Clusterware while attempting to stop the Oracle Clusterware stack.

  If any resources that Oracle Clusterware manages are still running after you run the `crsctl stop cluster` command, then the command fails. Use the `-f` parameter to unconditionally stop all resources and stop the Oracle Clusterware stack.
- If you intend to stop Oracle Clusterware on all or a list of nodes, then use the `crsctl stop cluster` command, because it prevents certain resources from being relocated to other servers in the cluster before the Oracle Clusterware stack is stopped on a particular server. If you must stop the Oracle High Availability

Services on one or more nodes, then wait until the `crsctl stop cluster` command completes and then run the `crsctl stop crs` command on any particular nodes, as necessary.

Example

To stop the Oracle Clusterware stack on a particular server:

```
# crsctl stop cluster -n node1
```

crsctl stop crs

Use the `crsctl stop crs` command to stop Oracle High Availability Services on the local server.

Syntax

```
crsctl stop crs [-f]
```

Usage Notes

- You must run this command as `root`.
- This command attempts to gracefully stop resources managed by Oracle Clusterware while attempting to stop Oracle High Availability Services on the local server.

  If any resources that Oracle Clusterware manages are still running after you run the `crsctl stop crs` command, then the command fails. Use the `-f` parameter to unconditionally stop all resources and stop Oracle High Availability Services on the local server.
- If you intend to stop Oracle Clusterware on all or a list of nodes, then use the `crsctl stop cluster` command, because it prevents certain resources from being relocated to other servers in the cluster before the Oracle Clusterware stack is stopped on a particular server. If you must stop the Oracle High Availability Services on one or more nodes, then wait until the `crsctl stop cluster` command completes and then run the `crsctl stop crs` command on any particular nodes, as necessary.
- Before attempting to shut down the Oracle Clusterware technology stack on all nodes with an Oracle ASM instance running in parallel in an Oracle Clusterware standard Cluster with Oracle Flex ASM enabled, you must first relocate at least one Oracle ASM instance to another node where Oracle ASM *is not* running.
- In Oracle Clusterware 11*g* release 2 (11.2.0.3), when you run this command in Solaris Sparc and Solaris X64 environments, drivers remain loaded on shutdown and subsequent startup. This *does not* happen in Linux environments.

Example

To stop Oracle High Availability Services on the local server:

```
# crsctl stop crs
```

crsctl stop ip

Use the `crsctl stop ip` command to stop a given IP name or IP address on a specified interface with a specified subnet mask. Run this command on the server on which you want to stop the IP.

Syntax

```
crsctl stop ip -A {IP_name | IP_address}/interface_name
```

Parameters

Table E–71 crsctl stop ip Command Parameters

Parameter	Description
{*IP_name* | *IP_address*}	Specify either a domain name or an IP address. If you do not specify a fully-qualified domain name, then CRSCTL uses a standard name search.
interface_name	Specify an interface on which to start the IP.

Example

To stop an IP on the local server, run the command similar to the following:

```
$ crsctl stop ip -A MyIP.domain.com/eth0
```

crsctl stop rollingpatch

The `crsctl stop rollingpatch` command transitions Oracle Clusterware and Oracle ASM out of rolling patch mode. Once transitioned out of rolling patch mode, the software does not tolerate nodes having different patch levels.

Syntax

```
crsctl stop rollingpatch
```

Usage Notes

- This command queries the Oracle Clusterware rolling patch state and Oracle ASM cluster state. If either one is in rolling patch mode, it will use the appropriate method to transition Oracle Clusterware or Oracle ASM out of rolling patch mode.
- This command verifies that all the nodes in the cluster have a consistent patch level, and returns an error otherwise.
- If neither Oracle Clusterware or Oracle ASM are in rolling patch mode when you issue this command, then this command does nothing.
- This command does not transition Oracle ASM out of rolling patch mode if issued within an Oracle ASM Client Cluster.

crsctl stop testdns

Use the `crsctl stop testdns` command to stop a test DNS server.

Syntax

```
crsctl stop testdns [-address address [-port port]] [-domain GNS_domain] [-v]
```

Parameters

Table E–72 crsctl stop testdns Command Parameters

Parameter	Description
`-address` *address*	Specify the server address for which you started the test DNS server in the form *IP_address*/*netmask* [/*interface_name*].
`-port` *port*	The port on which the DNS server listens. If you do not specify a port, then it defaults to port 53.
[`-domain` *GNS_domain*]	Domain for which the server should stop listening.
`-v`	Verbose output.

Example

The `crsctl stop testdns` command returns output similar to the following:

```
CRS-10032: successfully stopped the DNS listening daemon running on port 53 on
local node
```

crsctl unpin css

Use the `crsctl unpin css` command to unpin many servers. If a node is not pinned, its node number may change if the lease expires while it is down.

Syntax

```
crsctl unpin css -n node_name [node_name [...exit]]
```

Usage Notes

- You can specify a space-delimited list of nodes.
- Unpinned servers that stop for longer than a week are no longer reported by `olsnodes`. These servers are dynamic when they leave the cluster, so you do not need to explicitly remove them from the cluster.
- Deleting a node with the `crsctl delete node` command implicitly unpins the node.
- During upgrade of Oracle Clusterware, all servers are pinned, whereas after a fresh installation of Oracle Clusterware 12*c*, all servers you add to the cluster are unpinned.
- You *cannot* unpin a server that has an instance of Oracle RAC that is older than 12*c* release 1 (12.1) if you installed Oracle Clusterware 12*c* on that server.

Example

To unpin two nodes:

```
$ crsctl unpin css -n node1 node4
```

crsctl unset css

Use the `crsctl unset css` command to unset the value of a Cluster Synchronization Services parameter and restore it to its default value.

Syntax

```
crsctl unset css parameter
```

Usage Notes

- You can specify the following Cluster Synchronization Services parameters:
  - `diagwait`
  - `disktimeout`
  - `misscount`
  - `reboottime`
  - `priority`
  - `logfilesize`

Example

To restore the `reboottime` Cluster Synchronization Services parameter to its default value:

```
$ crsctl unset css reboottime
```

crsctl unset css ipmiconfig

Use the `crsctl unset css ipmiconfig` command to clear all previously stored IPMI configuration (login credentials and IP address) from the Oracle Local Registry. This is appropriate when deconfiguring IPMI in your cluster or if IPMI configuration was previously stored by the wrong user.

Syntax

```
crsctl unset css ipmiconfig
```

Usage Notes

- This command must be run under the user account originally used to configure IPMI or by a privileged user.
- This command only clears the IPMI configuration on the server on which you run it.
- If Oracle Clusterware was able to access and use the configuration data to be deleted by this command, then it will continue to do so until you restart Oracle Clusterware.

Example

To clear the IPMI configuration data from the Oracle Local Registry and restart Oracle Clusterware to prevent further use of IPMI, log in as `root` or a member of the Administrator's group on Windows and run the following commands:

```
crsctl unset css ipmiconfig
crsctl stop crs
crsctl start crs
```

crsctl unset css leafmisscount

Use the `crsctl unset css leafmisscount` command to clear the amount of time that passes before the grace time begins after communication fails between a Hub Node and a Leaf Node and reset to the default.

Syntax

```
crsctl unset css leafmisscount
```

Oracle Restart Environment CRSCTL Commands

The commands listed in this section control Oracle High Availability Services. These commands manage the Oracle High Availability Services stack in the Oracle Restart environment, which consists of the Oracle High Availability Services daemon (`ohasd`), Oracle ASM (if installed), and Cluster Synchronization Services (if Oracle ASM is installed). These commands only affect the local server on which you run them.

> **Note:** Oracle does not support using `crs_*` commands in an Oracle Restart environment.

Each server in the cluster is in one of two possible states:

- The whole stack is up, which means that Oracle High Availability Services is active
- The whole stack is down, which means that Oracle High Availability Services is inactive

You can use the following commands in the Oracle Restart environment, only:

- `crsctl check has`
- `crsctl config has`
- `crsctl disable has`
- `crsctl enable has`
- `crsctl query has releaseversion`
- `crsctl query has softwareversion`
- `crsctl start has`
- `crsctl stop has`

crsctl check has

Use the `crsctl check has` command to check the status of `ohasd`.

Syntax

```
crsctl check has
```

Example

The `crsctl check has` command returns output similar to the following:

```
CRS-4638: Oracle High Availability Services is online
```

crsctl config has

Use the `crsctl check has` command to display the automatic startup configuration of the Oracle High Availability Services stack on the server.

Syntax

```
crsctl config has
```

Example

The `crsctl config has` command returns output similar to the following:

```
CRS-4622 Oracle High Availability Services autostart is enabled.
```

crsctl disable has

Use the `crsctl disable has` command to disable automatic startup of the Oracle High Availability Services stack when the server boots up.

Syntax

```
crsctl disable has
```

Example

The `crsctl disable has` command returns output similar to the following:

```
CRS-4621 Oracle High Availability Services autostart is disabled.
```

crsctl enable has

Use the `crsctl enable has` command to enable automatic startup of the Oracle High Availability Services stack when the server boots up.

Syntax

```
crsctl enable has
```

Example

The `crsctl enable has` command returns output similar to the following:

```
CRS-4622 Oracle High Availability Services autostart is enabled.
```

crsctl query has releaseversion

Use the `crsctl query has releaseversion` command to display the release version of the Oracle Clusterware software that is stored in the binaries on the local node.

Syntax

```
crsctl query has releaseversion
```

Example

The `crsctl query has releaseversion` command returns output similar to the following:

```
Oracle High Availability Services release version on the local node is
[11.2.0.0.2]
```

crsctl query has softwareversion

Use the `crsctl query has softwareversion` command to display the software version on the local node.

Syntax

```
crsctl query has softwareversion
```

Usage Notes

- If you do not provide a server name, then Oracle Clusterware displays the version of Oracle Clusterware running on the local server.

Example

The `crsctl query has softwareversion` command returns output similar to the following:

```
Oracle High Availability Services version on the local node is [11.2.0.2.0]
```

crsctl start has

Use the `crsctl start has` command to start Oracle High Availability Services on the local server.

Syntax

```
crsctl start has [-noautostart]
```

Usage Notes

Use the `-noautorestart` parameter to start only Oracle High Availability Services.

Example

To start Oracle High Availability Services on the local server:

```
# crsctl start has
```

crsctl stop has

Use the `crsctl stop has` command to stop Oracle High Availability Services on the local server.

Syntax

```
crsctl stop has [-f]
```

Usage Notes

This command attempts to gracefully stop resources managed by Oracle Clusterware while attempting to stop Oracle High Availability Services.

If any resources that Oracle Clusterware manages are still running after you run the `crsctl stop has` command, then the command fails. Use the `-f` parameter to unconditionally stop all resources and stop Oracle High Availability Services.

Example

To stop Oracle High Availability Services on the local server:

```
# crsctl stop has
```

Troubleshooting and Diagnostic Output

You can use `crsctl set log` commands as the `root` user to enable dynamic debugging for Cluster Ready Services (CRS), Cluster Synchronization Services (CSS), and the Event Manager (EVM), and the clusterware subcomponents. You can dynamically change debugging levels using `crsctl debug` commands. Debugging information remains in the Oracle Cluster Registry (OCR) for use during the next startup. You can also enable debugging for resources.

This section covers the following topics:

- Dynamic Debugging Using crsctl set log
- Component Level Debugging
- Enabling Debugging for Oracle Clusterware Resources
- Enabling Additional Tracing for Oracle Clusterware Components

Dynamic Debugging Using crsctl set log

Use the `crsctl set log` command to set log levels for Oracle Clusterware.

Syntax

```
crsctl set log {[crs | css | evm "component_name=log_level, [...]"] |
[all=log_level]}
```

You can also set log levels for the agents of specific resources, as follows:

```
crsctl set log res "resource_name=log_level, [...]"
```

Usage Notes

- You can set log levels for various components of the three modules, CRS, CSS, and EVM. If you choose the `all` parameter, then you can set log levels for all components of one module with one command. Use the `crsctl lsmodules` command to obtain a list of components for each module.
- Enter a comma-delimited list of component name-log level pairs enclosed in double quotation marks (`""`).

> **Note:** Separate component name-log level pairs with an equals sign (=) in Oracle Clusterware 11*g* release 2 (11.2.0.3), and later. Previous Oracle Clusterware versions used a colon (`:`).

- The *`log_level`* is a number from `1` to `5` that sets the log level for the component or resource, where 1 is the least amount of log output and 5 provides the most detailed log output. The default log level is 2.
- To set log levels for resources, specify the name of a particular resource, or a comma-delimited list of resource name-log level pairs enclosed in double quotation marks (`""`).

Examples

To set log levels for the CRSRTI and CRSCOMM components of the CRS module:

```
$ crsctl set log crs "CRSRTI=1,CRSCOMM=2"
```

To set log levels for all components of the EVM module:

```
$ crsctl set log evm all=2
```

To set a log level for a resource:

```
$ crsctl set log res "myResource1=3"
```

Component Level Debugging

You can use `crsctl set log` and `crsctl set trace` commands as the `root` user to enable dynamic debugging for the various Oracle Clusterware modules.

Enabling Debugging for Oracle Clusterware Modules

You can enable debugging for Oracle Clusterware modules and their components, and for resources, by setting environment variables or by running `crsctl set log` commands, using the following syntax:

```
crsctl set {log | trace} module_name "component:debugging_level
[,component:debugging_level][,...]"
```

Run the `crsctl set` command as the `root` user, and supply the following information:

- *`module_name`*: The name of one of the following modules:

`mdns`: Multicast domain name server
`gpnp`: Grid Plug and Play service
`css`: Cluster Synchronization Services
`crf`: Cluster Health Monitor
`crs`: Cluster Ready Services
`ctss`: Cluster Time Synchronization Service
`evm`: Event Manager
`gipc`: Grid Interprocess Communication

- *`component`*: The name of a component for one of the modules. See Table E–73 for a list of all of the components.
- *`debugging_level`*: A number from `1` to `5` to indicate the level of detail you want the debug command to return, where 1 is the least amount of debugging output and 5 provides the most detailed debugging output. The default debugging level is 2.

The following commands show examples of how to enable debugging for the various modules:

- To enable debugging for Oracle Clusterware:

  ```
  crsctl set log crs "CRSRTI:1,CRSCOMM:2"
  ```

- To enable debugging for OCR:

  ```
  crsctl set log crs "CRSRTI:1,CRSCOMM:2,OCRSRV:4"
  ```

- To enable debugging for EVM:

  ```
  crsctl set log evm "EVMCOMM:1"
  ```

- To enable debugging for resources

  ```
  crsctl set log res "resname:1"
  ```

To obtain a list of components that can be used for debugging, run the `crsctl lsmodules` command, as follows:

```
crsctl lsmodules {mdns | gpnp | css | crf | crs | ctss | evm | gipc}
```

> **Note:** You do not have to be the `root` user to run the `crsctl lsmodulues` command.

Table E–73 shows the components for the CRS, CSS, and EVM modules, respectively. Note that some component names are common between the CRS, EVM, and CSS daemons and may be enabled on that specific daemon. For example, `COMMNS` is the NS layer and because each daemon uses the NS layer, you can enable this specific module component on any of the daemons to get specific debugging information.

Table E–73 Components for the CRS, CSS, and EVM Modules

CRS Components[1]	CSS Components[2]	EVM Components[3]
CRSUI	CSSD	EVMD
CRSCOMM	COMMCRS	EVMDMAIN
CRSRTI	COMMNS	EVMCOMM
CRSMAIN		EVMEVT
CRSPLACE		EVMAPP
CRSAPP		EVMAGENT
CRSRES		CRSOCR
CRSCOMM		CLUCLS
CRSOCR		CSSCLNT
CRSTIMER		COMMCRS
CRSEVT		COMMNS
CRSD		
CLUCLS		
CSSCLNT		
COMMCRS		
COMMNS		

[1] Obtain the list of CRS components using the `crsctl lsmodules crs` command.

[2] Obtain the list of CSS components using the `crsctl lsmodules css` command.

[3] Obtain the list of EVM components using the `crsctl lsmodules evm` command.

Example 1

To set debugging levels on specific cluster nodes, include the `-nodelist` keyword and the names of the nodes, as follows:

```
crsctl set log crs "CRSRTI:1,CRSCOMM:2" -nodelist node1,node2
```

Table E–74 describes the Cluster Synchronization Services modules.

Table E–74 Cluster Synchronization Services (CSS) Modules and Functions

Module	Description
CSS	CSS client component
CSSD	CSS daemon component

Table E–75 describes the function of each communication (COMM) module.

Table E–75 Communication (COMM) Modules and Functions

Module	Description
COMMCRS	Clusterware communication layer
COMMNS	NS communication layer

Table E–76 describes the functions performed by each CRS module.

Table E–76 Oracle Clusterware (CRS) Modules and Functions

Module	Descriptions
CRSUI	User interface module
CRSCOMM	Communication module
CRSRTI	Resource management module
CRSMAIN	Main module/driver
CRSPLACE	CRS placement module
CRSAPP	CRS application
CRSRES	CRS resources
CRSOCR	Oracle Cluster Registry interface
CRSTIMER	Various timers related to CRS
CRSEVT	CRS EVM/event interface module
CRSD	CRS daemon

Using the `crsctl set log crs` command, you can debug the OCR components listed in Table E–77. The components listed in Table E–77 can also be used for the Oracle Local Registry (OLR) except for OCRMAS and OCRASM. You can also use them for OCR and OLR clients, except for OCRMAS and OCRSRV. Some OCR and OLR clients are OCRCONFIG, OCRDUMP, and so on.

Table E–77 Oracle Cluster Registry (OCR) Component Names

Module	Description
OCRAPI	OCR abstraction component
OCRCLI	OCR client component
OCRSRV	OCR server component
OCRMAS	OCR master thread component
OCRMSG	OCR message component
OCRCAC	OCR cache component
OCRRAW	OCR raw device component
OCRUTL	OCR util component
OCROSD	OCR operating system dependent (OSD) layer
OCRASM	OCR ASM component

Table E–78 describes the OCR tool modules.

Table E–78 OCRCONFIG Modules and Functions

Module	Description
OCRCONFIG	OCRCONFIG component for configuring OCR
OCRDUMP	OCRDUMP component that lists the Oracle Cluster Registry contents
OCRCHECK	OCRCHECK component that verifies all of the configured OCRs

Enabling Debugging for Oracle Clusterware Resources

You can enable debugging for Oracle Clusterware resources by running the `crsctl set log` command, using the following syntax:

```
crsctl set log res "resource_name=debugging_level"
```

Run the `crsctl set log` command as the `root` user, and supply the following information:

- *`resource_name`*: The name of the resource to debug.
- *`debugging_level`*: A number from `1` to `5` to indicate the level of detail you want the debug command to return, where 1 is the least amount of debugging output and 5 provides the most detailed debugging output. The default debugging level is 2.

To obtain a list of resources that can be used for debugging, run the `crsctl status resource` command.

Example

To generate a debugging log for the VIP resource on `node1`, issue the following command:

```
crsctl set log res "ora.node1.vip:1"
```

Enabling Additional Tracing for Oracle Clusterware Components

My Oracle Support may ask you to enable tracing to capture additional information. Because the procedures described in this section may affect performance, only perform these activities with the assistance of My Oracle Support.

You can enable tracing for Oracle Clusterware resources by running the `crsctl set trace` command, using the following syntax:

```
crsctl set trace module_name "component_name=tracing_level,..."
```

Run the `crsctl set trace` command as the `root` user, and supply the following information:

- *`module_name`*: The name of one of the following modules:

`mdns`: Multicast domain name server
`gpnp`: Grid Plug and Play service
`css`: Cluster Synchronization Services
`crf`: Cluster Health Monitor
`crs`: Cluster Ready Services
`ctss`: Cluster Time Synchronization Service
`evm`: Event Manager
`gipc`: Grid Interprocess Communication

- *component_name*: The name of the component for one of the modules. See Table E–73 for a list of components.
- *tracing_level*: A number from `1` to `5` to indicate the level of detail you want the trace command to return, where 1 is the least amount of tracing output and 5 provides the most detailed tracing output.

Example 1

To generate a trace file for Cluster Synchronization Services, use the following command:

```
crsctl set trace "css=3"
```

F

Rapid Home Provisioning and Server Control Command Reference

This appendix contains reference information for Rapid Home Provisioning commands, including the Rapid Home Provisioning Control (RHPCTL) utility and Server Control (SRVCTL) utility.

This appendix includes the following topics:

- RHPCTL Command Reference
- SRVCTL Command Reference

RHPCTL Command Reference

This section describes RHPCTL command usage information, and lists and describes RHPCTL commands.

RHPCTL Overview

RHPCTL is a command-line utility with which you perform Rapid Home Provisioning operations and manage Rapid Home Provisioning Servers and Clients. RHPCTL uses the following syntax:

```
rhpctl command object [parameters]
```

In RHPCTL syntax:

- *`command`* is a verb such as `add`, `delete`, or `query`
- *`object`* (also known as a noun) is the target or object on which RHPCTL performs the command, such as client or image.
- *`parameters`* extend the use of a preceding command combination to include additional parameters for the command. Specify parameters as `-`*`keyword value`*. If the *`value`* field contains a comma-delimited list, then do not use spaces between the items in the list.

You can use RHPCTL commands to perform several Rapid Home Provisioning operations, including:

- Rapid Home Provisioning Client operations, such as creating a Rapid Home Provisioning Client configuration.
- Role operations, such as adding and deleting roles, and granting and revoking roles for users.

- Site operations, such as obtaining configuration information for Rapid Home Provisioning Servers.
- Image operations, such as adding, deleting, and importing images.
- Image series operations, such as adding and deleting image series.
- Working copy operations, such as adding and deleting working copies.

Using RHPCTL Help

To see help for all RHPCTL commands, from the command line enter:

```
rhpctl -help
```

To see the command syntax and a list of parameters for each RHPCTL command, from the command line enter:

```
rhpctl command (or verb) object (or noun) -help
```

rhpctl add client

Use the `rhpctl add client` command to add a Rapid Home Provisioning Client to the configuration.

Syntax

```
rhpctl add client -client client_name -toclientdata path [-maproles
    role=user_name[,role=user_name[,...]]]
```

Parameters

Table F–1 rhpctl add client Parameters

Parameter	Description
`-client` *`client_name`*	Specify the name of the cluster in which you want to create the client.
`-toclientdata` *`path`*	Specify the path to the XML file that is created by the Rapid Home Provisioning Server (specific to the client cluster), which contains the information the client needs to configure its connection to the server.

`-maproles` *`role`*`=`
`user_name``[,...]`
	You can specify either built-in roles or roles that you have defined.

Usage Notes

- You can only run this command on the Rapid Home Provisioning Server

Example

To add a client to the Rapid Home Provisioning Server:

```
$ rhpctl add client -client ClientCluster3 -toclientdata Grid_home/RHPserver/info
```

rhpctl add database

Use the `rhpctl add database` command to add a database using a specific working copy.

Syntax

```
rhpctl add database -workingcopy workingcopy_name -dbname unique_db_name
   -datafileDestination datafileDestination_path {-node node_list | -serverpool
pool_name
   [-pqpool pool_name | -newpqpool pool_name -pqcardinality cardinality]
   | -newpool pool_name -cardinality cardinality [-pqpool pool_name
   | -newpqpool pool_name -pqcardinality cardinality]} [-dbtype
   {RACONENODE | RAC | SINGLE}] [-dbtemplate file_path | image_name:relative_file_
path]
```

Parameters

Table F–2 rhpctl add database Parameters

Parameter	Description
`-workingcopy` *`workingcopy_name`*	Specify the name of an existing working copy for the database that you want to add.
`-dbname` *`unique_db_name`*	Specify the unique name of the database (`DB_UNIQUE_NAME`) that you are adding.
`-datafileDestination` *`datafileDestination_path`*	Specify the data file destination location or the name of the Oracle Automatic Storage Management (Oracle ASM) disk group. **Note:** You cannot specify a disk group for Oracle Database versions before Oracle Database 11*g* release 2 (11.2).
`-node` *`node_list`*	Specify a node or comma-delimited list of several nodes on which to create the database.
`-serverpool` *`server_pool_name`*	Specify the name of an existing server pool.
`-pqpool` *`server_pool_name`*	Specify the name of an existing server pool. **Note:** This parameter is only applicable in an Oracle Flex Cluster environment and refers to server pools (either already defined, as in this case, or to be created when you use the `-newpqpool` parameter) running on Leaf Nodes.
`-newpqpool` *`server_pool_name`*	Optionally, you can create a new server pool to be used for parallel queries. Specify a name for the new server pool. **Note:** This parameter is only applicable in an Oracle Flex Cluster environment because it refers to server pools running on Leaf Nodes.
`-pqcardinality` *`cardinality`*	If you create a new server pool, then you must specify a cardinality value for the server pool. **Note:** This parameter is only applicable in an Oracle Flex Cluster environment.
`-newpool` *`server_pool_name`*	Optionally, you can create a new server pool. Specify a name for the new server pool.
`-cardinality` *`cardinality`*	If you create a new server pool, then you must specify a cardinality value for the server pool.
`-dbtype {RACONENODE | RAC | SINGLE}`	Specify whether the database is Oracle RAC One Node, Oracle RAC, or a nonclustered database.
`-dbtemplate` *`file_path`* `|` *`image_name`*`:`*`relative_file_path`*	Specify the absolute file path to a database template or the relative path to the image home directory on a Rapid Home Provisioning Server.

Example

To add a database from a working copy named prodhome:

```
$ rhpctl add database -workingcopy prodhome -dbname proddb -datafileDestination
/acfs/proddata -serverpool prodpool1 -dbtype RAC
```

rhpctl add image

Use the `rhpctl add image` command to create an image from an existing working copy and add it to the list of existing images on the Rapid Home Provisioning Server configuration.

Syntax

```
rhpctl add image -image image_name -workingcopy working_copy_name
   [-series series_name] [-state {TESTABLE | RESTRICTED | PUBLISHED}]
```

Parameters

Table F–3 rhpctl add image Parameters

Parameter	Description
`-image` *`image_name`*	Specify the name of the image that you want to add.
`-workingcopy` *`working_copy_name`*	Specify the name of the working copy from which to create the image.
`-series` *`series_name`*	Specify the name of the series.
`-state {TESTABLE	RESTRICTED	PUBLISHED`	Specify the state of the image.

rhpctl add role

Use the `rhpctl add role` command to create roles and add them to the list of existing roles on the Rapid Home Provisioning Server configuration.

> **See Also:** "Rapid Home Provisioning Roles" on page 5-3

Syntax

```
rhpctl add role -role role_name -hasRoles roles
```

Parameters

Table F–4 rhpctl add role Parameters

Parameter	Description
`-role` *`role_name`*	Specify a name for the role that you want to create.

Table F–4 (Cont.) rhpctl add role Parameters

Parameter	Description
`-hasRoles` *roles*	Specify a comma-delimited list of roles to include with the new role. GH_ROLE_ADMIN GH_SITE_ADMIN GH_SERIES_ADMIN GH_SERIES_CONTRIB GH_WC_ADMIN GH_WC_USER GH_IMG_ADMIN GH_IMG_USER GH_IMG_TESTABLE GH_IMG_RESTRICT GH_IMG_PUBLISH GH_IMG_VISIBILITY GH_SA GH_CA GH_OPER

Usage Notes

- You can only run this command on the Rapid Home Provisioning Server
- You must be assigned the GH_ROLE_ADMIN role to run this command

Example

To add a role on the Rapid Home Provisioning Server:

```
$ rhpctl add role -role hr_admin -hasRoles GH_WC_USER,GH_IMG_USER
```

rhpctl add series

Use the `rhpctl add series` command to add a series to the Rapid Home Provisioning Server configuration.

Syntax

```
rhpctl add series -series series_name [-image image_name]]
```

Parameters

Table F–5 rhpctl add series Parameters

Parameter	Description
`-series` *series_name*	Specify a name for the series that you want to add.
`-image` *image_name*	Specify the name of a configured image. This image becomes the first in the series.

rhpctl add workingcopy

Use the `rhpctl add workingcopy` command to create a working copy.

Syntax

```
rhpctl add workingcopy -workingcopy workingcopy_name -image image_name
   [-oraclebase oraclebase_path] [-path absolute_path] [-storagetype {NFS | LOCAL}
   [-user user_name] [-dbname unique_db_name [-dbtype {RACONENODE | RAC | SINGLE}]
   [-datafileDestination datafileDestination_path] [-dbtemplate file_path |
   image_name:relative_file_path
   {-node node_list | -serverpool server_pool_name [-pqpool pool_name
   | -newpqpool pool_name -pqcardinality cardinality] | -newpool server_pool_name
    -cardinality cardinality [-pqpool pool_name | -newpqpool pool_name
    -pqcardinality cardinality]} [-cdb] [-pdbprefix pdb_prefix [-pdbcount pdb_
count]]]
   [-client cluster_name] [-ignoreprereq]
```

Parameters

Table F–6 rhpctl add workingcopy Parameters

Parameter	Description
`-workingcopy` *`workingcopy_name`*	Specify a name for the working copy that you want to create.
`-image` *`image_name`*	Specify the name of a configured image from which to create a working copy.
`-oraclebase` *`oraclebase_path`*	Specify an `ORACLE_BASE` path for provisioning an Oracle Database home. You can specify either an existing directory or a new directory. **Note:** This parameter is required only for the `ORACLEDBSOFTWARE` image type.
`-path` *`absolute_path`*	Specify the absolute path for provisioning the software home on the client side (this location must be empty). For Oracle Database images, this becomes the `ORACLE_HOME`. **Note:** You cannot use this parameter if you are provisioning a working copy to a Rapid Home Provisioning Server because, in that case, Oracle ACFS creates a snapshot path within the file system.
`-storagetype {NFS | LOCAL}`	Specify the type of storage for the software home. The default is `NFS`.
`-user` *`user_name`*	Specify the name of the user that will own the working copy being provisioned. If you do not specify this parameter, then the working copy is owned by the user running the command. If you are provisioning to a remote cluster, then the user name must be a valid user on the remote cluster. The user ID need not be the same between the two clusters, but the user name must exist on both.
`-dbname` *`unique_db_name`*	Specify the unique name of the database (`DB_UNIQUE_NAME`) that you are provisioning.
`-dbtype {RACONENODE | RAC | SINGLE}`	Specify whether the database is Oracle RAC One Node, Oracle RAC, or single instance (non-Oracle RAC).
`-datafileDestination` *`datafileDestination_path`*	Specify the data file destination location or the name of the Oracle Automatic Storage Management (Oracle ASM) disk group. **Notes:** ■ This defaults to `ORACLE_BASE/oradata` when you specify `SINGLE` for `-dbtype`. ■ This parameter is required only for the `ORACLEDBSOFTWARE` image type. ■ You cannot specify an Oracle ASM disk group for Oracle Databases older than 11*g* release 2 (11.2)

Table F–6 (Cont.) rhpctl add workingcopy Parameters

Parameter	Description
-dbtemplate *file_path* | *image_name*:*relative_file_path*	Specify the absolute path to a database template or the relative path to the image home directory on the Rapid Home Provisioning Server. If you do not specify a database template, then RHPCTL uses the default template.
-node *node_list*	Specify a node or comma-delimited list of several nodes on which to create the database. **Note:** This parameter is required when the value of -dbtype is SINGLE.
-serverpool *server_pool_name*	Specify the name of an existing server pool. **Note:** This defaults to the name of the working copy.
-newpool *server_pool_name*	Optionally, you can create a new server pool. Specify a name for the new server pool.
-cardinality *cardinality*	If you create a new server pool, then you must specify a cardinality value for the server pool.
-pqpool *server_pool_name*	Specify the name of an existing server pool. **Note:** This parameter is only applicable in an Oracle Flex Cluster environment and refers to server pools (either already defined as in this case, or to be created when you use the -newpqpool parameter) running on Leaf Nodes.
-newpqpool *server_pool_name*	Optionally, you can create a new server pool. Specify a name for the new server pool. **Note:** This parameter is only applicable in an Oracle Flex Cluster environment because it refers to server pools running on Leaf Nodes.
-pqcardinality *cardinality*	If you create a new server pool, then you must specify a cardinality value for the server pool. **Note:** This parameter is only applicable in an Oracle Flex Cluster environment.
-cdb	Use this parameter if you want to create the database as a container database (CDB).
-pdbprefix *pdb_prefix*	Specify the pluggable database (PDB) prefix if you are creating one or more PDBs.
-pdbcount *pdb_count*	Specify the number of PDBs you want to create.
-client *cluster_name*	Specify the name of the client cluster.

Usage Notes

You can obtain context sensitive help for specific use cases for the rhpctl add workingcopy command, as follows:

```
$ rhpctl add workingcopy -help [REMOTEPROVISIONING | STORAGETYPE | ADMINDB
 | POLICYDB | DBWITHPQPOOLS | DBTEMPLATE | PDB]
```

Examples

The following example provisions an Oracle Database software home from a gold image:

```
$ rhpctl add workingcopy -workingcopy workingcopy_name -image image_name
 -oraclebase oraclebase_path
```

The following example provisions an Oracle Database software home and creates a policy-managed database from a gold image:

```
$ rhpctl add workingcopy -workingcopy workingcopy_name -image image_name
 -oraclebase oraclebase_path -dbname unique_db_name -datafileDestination
 datafileDestination_path -serverpool pool_name
```

The following example provisions an Oracle Database software home on a user-specified storage location and creates a policy-managed database:

```
$ rhpctl add workingcopy -workingcopy workingcopy_name -image image_name
 -oraclebase oraclebase_path -storagetype LOCAL -path storage_path -dbname
 unique_db_name -datafileDestination datafileDestination_path -newpool
 server_pool_name -cardinality cardinality
```

> **Note:** If you are provisioning Oracle database software to a Rapid Home Provisioning Client that has been configured with a disk group, then do not specify the `-path` parameter, so as to enable the Rapid Home Provisioning Client to use storage provided by Rapid Home Provisioning.
>
> If the Rapid Home Provisioning Client is not configured with a disk group, then specify the `-storagetype` parameter with either `NFS` or `LOCAL`, in addition to specifying the `-path` parameter.

rhpctl allow

This section describes the `rhpctl allow image` command.

rhpctl allow image

Use the `rhpctl allow image` command to allow access to an image by a user or a role.

Syntax `rhpctl allow image -image image_name {-user user_name [-client cluster_name] | -role role_name}`

Parameters

Table F–7 rhpctl allow image Parameters

Parameter	Description
`-image image_name`	Specify the name of the image to which you want to allow access.

`-user user_name`
`-client cluster_name |`
`-role role_name`
	Specify the either of the following:
	▪ A user for which you want to allow access to the image and, optionally, the cluster name of the client cluster with the user.
	▪ The role for which you want to allow access to the image.

rhpctl delete client

Use the `rhpctl delete client` command to delete a specific Rapid Home Provisioning Client from the configuration.

Syntax

```
rhpctl delete client -client cluster_name [-force]
```

Usage Notes

- You must stop the Rapid Home Provisioning Client before you run this command or use the `-force` option
- Specify the name of the client cluster that you want to delete from the configuration

rhpctl delete database

Use the `rhpctl delete database` command to delete a database that was created from a working copy.

Syntax

```
rhpctl delete database -workingcopy workingcopy_name -dbname unique_db_name
```

Parameters

Table F–8 rhpctl delete database Parameters

Parameter	Description
`-workingcopy` *`workingcopy_name`*	Specify a name for the working copy for the database that you want to delete.
`-dbname` *`unique_db_name`*	Specify the unique name of the database (`DB_UNIQUE_NAME`) that you are deleting.

Example

To delete a database from a working copy:

```
$ rhpctl delete database -workingcopy -dbname hr1
```

rhpctl delete image

Use the `rhpctl delete image` command to delete a specific image.

Syntax

```
rhpctl delete image -image image_name
```

Usage Notes

- Specify the name of the image you want to delete.

Example

The following example deletes an image named PRODIMAGEV0:

```
$ rhpctl delete image -image PRODIMAGEV0
```

rhpctl delete role

Use the `rhpctl delete role` command to delete a role from the list of existing roles on the Rapid Home Provisioning Server configuration.

Syntax

```
rhpctl delete role -role role_name
```

Usage Notes

- Specify the name of the role that you want to delete
- You can only run this command on the Rapid Home Provisioning Server

Example

To delete a role from the Rapid Home Provisioning Server:

```
$ rhpctl delete role -role hr_admin
```

rhpctl delete series

Use the `rhpctl delete series` command to delete a series from the Rapid Home Provisioning Server configuration.

Syntax

```
rhpctl delete series -series series_name [-force]]
```

Parameters

Table F–9 rhpctl delete series Parameters

Parameter	Description
`-series` *`series_name`*	Specify the name of the series that you want to delete
`-force`	Use `-force` to delete an image series even if the series includes images.

Usage Notes

- Before deleting an image series, you must first remove all images from the series by using the `rhpctl deleteimage series` command.

Example

The following example deletes a series called PRODDBSERIES:

```
$ rhpctl delete series -series PRODDBSERIES
```

rhpctl delete user

Use the `rhpctl delete user` command to delete a user from the Rapid Home Provisioning repository.

Syntax

```
rhpctl delete user -user user_name [-client cluster_name]
```

Parameters

Table F–10 rhpctl delete user Parameters

Parameter	Description
`-user` *`user_name`*	Specify the name of the user you want to delete from a Rapid Home Provisioning Client.
`-client` *`cluster_name`*	Optionally, you can specify the name of the client cluster from which you want to delete from a specific user.

Usage Notes

- You can delete non-built-in users only if that user does not own any working copies.
- If the user created an image or image series, then you can still delete the user, but the creator of the image or image series is changed to *internal-user@GHS*.
- If the user was the owner of an image series, then you can delete the user, but the owner of the image series will be changed to *internal-user@GHS*. You can still use the affected image series as normal, such that you can still provision a working copy from the affected image series, and you can still insert or delete images from the affected image series.

Example

The following example deletes the user named `mjkeenan` on the server cluster from the Rapid Home Provisioning repository:

```
$ rhpctl delete user -user mjkeenan
```

rhpctl delete workingcopy

Use the `rhpctl delete workingcopy` command to delete an existing working copy.

Syntax

```
rhpctl delete workingcopy -workingcopy workingcopy_name [-force]
```

Usage Notes

- Specify the name of a working copy that you want to delete.
- Use the `-force` option to forcibly delete a working copy.
- This command *does not* delete the Oracle base that was created when you ran `rhpctl add workingcopy`.

rhpctl deleteimage series

Use the `rhpctl deleteimage series` command to delete an image from a series.

Syntax

```
rhpctl deleteimage series -series series_name -image image_name
```

Parameters

Table F–11 rhpctl deleteimage series Parameters

Parameter	Description
`-series` *`series_name`*	Specify the name of the series from which you want to delete an image.
`-image` *`image_name`*	Specify the name of the image that you want to delete from a series.

Example

The following command deletes an image called PRODIMAGEV0 from a series called PRODDBSERIES:

```
$ rhpctl deleteimage series -series PRODDBSERIES -image PRODIMAGEV0
```

rhpctl disallow image

Use the `rhpctl disallow image` command to disallow access to an image by a user or a role.

Syntax

```
rhpctl disallow image -image image_name {-user user_name [-client client_name]
    | -role role_name}
```

Parameters

Table F–12 rhpctl disallow image Parameters

Parameter	Description
`-image` *`image_name`*	Specify the name of the image to which you want to disallow access.

`-user` *`user_name`*
`-client` *`cluster_name`* |
`-role` *`role_name`*
	Specify either of the following:
	■ A user for which you want to disallow access to the image and, optionally, the cluster name of the client cluster with the user.
	■ The role for which you want to disallow access to the image.

Example

To disallow access to an image:

```
$ rhpctl disallow image -image PRODIMAGE -user mjk -client GHC1
```

rhpctl export client

Use the `rhpctl export client` command to export data from the repository on the Rapid Home Provisioning Server to a client data file.

Syntax

```
rhpctl export client -client cluster_name -clientdata file_path
```

Parameters

Table F–13 rhpctl export client Parameters

Parameter	Description
`-client` *`cluster_name`*	Specify the name of the client cluster that you want to export.
`-clientdata` *`file_path`*	Specify the path to the location of the client data file.

Usage Notes

- You can only run this command on the Rapid Home Provisioning Server

Example

To export repository data from a Rapid Home Provisioning Client named `mjk9394` to a client data file, `/tmp/mjk9394.xml`:

```
$ rhpctl export client -client mjk9394 -clientdata /tmp/mjk9394.xml
```

rhpctl grant role

Use the `rhpctl grant role` command to grant a role to a client user or to another role.

Syntax

```
rhpctl grant role {-role role_name {-user user_name [-client cluster_name]
   | -grantee role_name}}
   | {[-client cluster_name] [-maproles role=user_name[+user_name...]
   [,role=user_name[+user_name...]...]}
```

Parameters

Table F–14 rhpctl grant role Parameters

Parameter	Description
`-role` *`role_name`*	Specify the name of the role that you want to grant clients or users.

`-user` *`user_name`*
`-client` *`cluster_name`*
	Specify the name of a user and, optionally, a client cluster to which you want to grant a role. The user name that you specify must be in the form of *`user@rhpclient`*, where *`rhpclient`* is the name of the Rapid Home Provisioning Client.

`-grantee` *`role_name`*	Use this parameter to specify a role to which you want to grant another role.

`[-client` *`client_name`*`]`
`-maproles` *`role=`* *`user_name`*`[+`*`user_name`*`...]`
	You can map either built-in roles or roles that you have defined to specific users. Use a plus sign (+) to map more than one user to a specific role. Separate additional role/user pairs with commas. Optionally, you can also specify a client cluster.

Example

The following example grants a role, ABC, to another role, XYZ:

```
$ rhpctl grant role -role ABC -grantee XYZ
```

rhpctl import image

Use the `rhpctl import image` command to create an image by copying the entire software contents from the specified path to the Rapid Home Provisioning Server.

Syntax

```
rhpctl import image -image image_name -path path_to_installed_home
   [-imagetype ORACLEDBSOFTWARE | SOFTWARE] [-owner user_name]
   [-state {TESTABLE | RESTRICTED | PUBLISHED}]
```

Parameters

Table F–15 rhpctl import image Parameters

Parameter	Description
`-image` *`image_name`*	Specify the name of the image that you want to add.
`-path` *`path_to_installed_home`*	Specify the absolute path location of the software home that you want to import (for Oracle Database images, this is the `ORACLE_HOME`).

Table F–15 (Cont.) rhpctl import image Parameters

Parameter	Description
`-imagetype ORACLEDBSOFTWARE|SOFTWARE`	Specify the type of image you are importing. If you are using Oracle Database software, then specify `ORACLEDBSOFTWARE`. Otherwise, specify `SOFTWARE`.
`-pathowner` *`user_name`*	Specify the user with read access to the files and directories under the specified path. **Note:** This parameter is applicable only for non-Oracle database software homes.
`-state {TESTABLE | RESTRICTED | PUBLISHED`	Specify whether the state of the image is testable, restricted, or published.

Example

To import an image:

```
$ rhpctl import image -image PRODIMAGEV1 -path /u01/app/product/12.1.0/dbhome
-pathowner orcl
```

rhpctl insertimage series

Use the `rhpctl insertimage series` command to insert a new image into a series.

Syntax

```
rhpctl insertimage series -series series_name -image image_name
   [-before image_name]
```

Parameters

Table F–16 rhpctl insertimage series Parameters

Parameter	Description
`-series` *`series_name`*	Specify the name of the series into which you want to insert an image.
`-image` *`image_name`*	Specify the name of the image that you want to insert into a series.
`-before` *`image_name`*	Optionally, you can specify the name of an image before which you want to insert the new image.

rhpctl move database

Use the `rhpctl move database` command to move one or more databases from a source working copy or any Oracle home to a patched working copy. The patched working copy need not exist and can be created dynamically, in which case you must specify an image name from which to create the patched working copy.

Syntax

```
rhpctl move database {-sourcewc workingcopy_name | -sourcehome Oracle_home_path
   [-oraclebase Oracle_base_path] [-client cluster_name]} -patchedwc workingcopy_
name
   [-dbname unique_db_name] [-nonrolling] [-image image_name] [-path where_path]
```

Parameters

Table F–17 rhpctl move database Parameters

Parameter	Description
-sourcewc *workingcopy_name*	Specify the name of the working copy from which the database is to be moved.
-sourcehome *Oracle_home_path*	Specify the source Oracle home path.
-oraclebase *Oracle_base_path*	Specify the ORACLE_BASE path for provisioning the Oracle database home (required only for ORACLEDBSOFTWARE image type).
-client *cluster_name*	Specify the name of the client cluster.
-patchedwc *workingcopy_name*	Specify the name of the working copy to where you want to move the database.
-dbname *unique_db_name*	Specify the unique name of the database (DB_UNIQUE_NAME) that you want to provision.
-nonrolling	Specify this parameter to move the database in a non-rolling mode. By default, databases move in a rolling mode.
-image *image_name*	Optionally, you can specify the name of the database software image you want to move.
-path *where_path*	Optionally, you can specify the directory path to where you want to move the database.

Example

To move all the databases running from one working copy to another in a rolling fashion:

```
rhpctl move database -sourcewc prodHomeV1 -patchedwc prodHomeV2 -client
prodcluster -dbname prod1db
```

In the preceding example, the patched working copy, prodHomeV2, must exist.

rhpctl modify client

Use the `rhpctl modify client` command to modify a client.

Syntax

```
rhpctl modify client -client cluster_name [-enabled {TRUE | FALSE}]
   [-maproles role=user_name[+user_name...][,role=user_name[+user_name...]...]
   [-password]]
```

Parameters

Table F–18 rhpctl modify client Parameters

Parameter	Description
-client *cluster_name*	Specify the name of the client cluster that you want to modify.
-enabled {TRUE | FALSE}	Specify whether the client is enabled.

Table F–18 (Cont.) rhpctl modify client Parameters

Parameter	Description
-maproles *role*= *user_name*[+*user_name*...]	You can map either built-in roles or roles that you have defined to specific users. Use a plus sign (+) to map more than one user to a specific role. Separate additional role/user pairs with commas.
-password	Optionally, you can modify the password for the client cluster.

rhpctl promote image

Use the `rhpctl promote image` command to promote an image.

Syntax

```
rhpctl promote image -image image_name -state {TESTABLE | RESTRICTED | PUBLISHED}
```

Parameters

Table F–19 rhpctl promote image Parameters

Parameter	Description
-image *image_name*	Specify the name of the image that you want to promote.
-state {TESTABLE | RESTRICTED | PUBLISHED}	Specify one of the following as the name of the state of the image: TESTABLE: RESTRICTED: PUBLISHED:

rhpctl query client

Use the `rhpctl query client` command to display configuration information of a specific Rapid Home Provisioning Client cluster.

Syntax

```
rhpctl query client [-client cluster_name]
```

Usage Notes

- Specify the name of the client cluster in which the Rapid Home Provisioning Client resides for which you want to display the configuration information

rhpctl query image

Use the `rhpctl query image` command to display the configuration information of an existing image.

Syntax

```
rhpctl query image [-image image_name [-dbtemplate]]
```

Parameters

Table F–20 rhpctl query image Parameters

Parameter	Description
`-image` *`image_name`*	Specify the name of the image for which you want to display the configuration.
`-dbtemplate`	

rhpctl query role

Use the `rhpctl query role` command to display the configuration information of a specific role.

Syntax

```
rhpctl query role -role role_name
```

Usage Notes

- Specify the name of the role for which you want to display the configuration information
- You can only run this command on the Rapid Home Provisioning Server

rhpctl query series

Use the `rhpctl query series` command to display the configuration of a series.

Syntax

```
rhpctl query series [-series series_name | [-image image_name]]
```

Parameters

Table F–21 rhpctl query series Parameters

Parameter	Description
`-series` *`series_name`*	Specify the name of the series for which you want to display the configuration.
`-image` *`image_name`*	Alternatively, you can specify the name of a configured image.

rhpctl query server

Use the `rhpctl query server` command to display the configuration of a server.

Syntax

```
rhpctl query server]
```

rhpctl query workingcopy

Use the `rhpctl query workingcopy` command to display the configuration information of an existing working copy.

Syntax

```
rhpctl query workingcopy [-workingcopy workingcopy_name | -image image_name]
```

Parameters

Table F–22 rhpctl query workingcopy Parameters

Parameter	Description
-workingcopy *workingcopy_name*	Specify the name of a working copy for which you want to display the configuration information.
-image *image_name*	Alternatively, you can specify the name of a configured image you want to query.

rhpctl revoke role

Use the `rhpctl revoke role` command to revoke a role from a client user.

Syntax

```
rhpctl revoke role -role role_name {-user user_name [-client cluster_name]
   | -grantee role_name}
   | {[-client cluster_name] [-maproles role=user_name[+user_name...]
   [,role=user_name[+user_name...]...]}
```

Parameters

Table F–23 rhpctl revoke role Parameters

Parameter	Description
-role *role_name*	Specify the name of the role from which you want to revoke clients or users.
-user *user_name* -client *cluster_name*	Specify the name of a user and, optionally, a client cluster from which you want to revoke a role. The user name that you specify must be in the form of *user@rhpclient*, where *rhpclient* is the name of the Rapid Home Provisioning Client.
-grantee *role_name*	Specify the name of the role.
[-client *client_name*] -maproles *role*= *user_name*[+*user_name*...]	You can map either built-in roles or roles that you have defined to specific users. Use a plus sign (+) to map more than one user to a specific role. Separate additional role/user pairs with commas. Optionally, you can also specify a client cluster.

SRVCTL Command Reference

This section lists and describes all SRVCTL commands related to Rapid Home Provisioning.

> **See Also:** *Oracle Real Application Clusters Administration and Deployment Guide* for SRVCTL command usage information

srvctl add asm

Adds a record for an Oracle ASM instance to the entire cluster. This command must be run only one time from the Oracle Grid Infrastructure home.

Notes:

- You can only use this command and the parameters described with Oracle Clusterware.
- To manage Oracle ASM on Oracle Database 12*c* installations, use the SRVCTL binary in the Oracle Grid Infrastructure home for a cluster (Grid home). If you have Oracle RAC or Oracle Database installed, then you cannot use the SRVCTL binary in the database home to manage Oracle ASM.

See Also: *Oracle Database Administrator's Guide* for other command parameters that you can use with Oracle Restart

Syntax and Parameters

Use the `srvctl add asm` command with the following syntax:

```
srvctl add asm [-listener listener_name] [-pwfile password_file_path
    [-remote [-count {number_of_instances | ALL}] | -proxy]
```

Table F–24 srvctl add asm Parameters

Parameter	Description
`-listener listener_name`	Enter the name of a listener. If you do not specify this parameter, then the listener name defaults to `LISTENER`.
`-pwfile password_file_path`	Enter the full path to the location of the password file on Oracle ASM.
`-remote [-count {number_of_instances |ALL}]	-proxy]`	Indicates that the Oracle ASM instances are configured as Oracle Flex ASM instances. You can optionally specify the number of Oracle ASM instances to create or that an Oracle ASM instance should be created on all nodes. Additionally, you can specify the `-proxy` parameter to configure the Oracle ASM Dynamic Volume Manager proxy instance resource.

Example To add a clusterware resource for Oracle ASM to every node in the cluster, use the following command:

```
$ srvctl add asm
```

srvctl add exportfs

Creates an export file system configuration in Oracle Clusterware.

Note: This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl add exportfs` command with the following syntax:

```
srvctl add exportfs -name unique_name -id havip_name -path path_to_export
    [-clients nfs_client_string] [-options nfs_client_string]
```

Table F–25 srvctl add exportfs Parameters

Parameter	Description
`-name` *`unique_name`*	Enter a unique name for the NFS export you are creating. This parameter is required.
`-id` *`havip_name`*	Enter the number or name that identifies an existing HAVIP, that you specified in the `srvctl add havip` command. This parameter is required.
`-path` *`path_to_export`*	Enter the path to the Oracle Automatic Storage Management Cluster File System (Oracle ACFS) that you want to export. This parameter is required.

`-clients` *`nfs_client_string`*
`-options` *`nfs_client_string`*
	Optionally, enter a comma-delimited list of clients to which you want the file system exported. You can enter net groups and wildcards. For example:
	`-clients @netgroup1,`
	`*.us.domain.com,`
	`16.140.0.0, hostname1`
	Note: You can use the `-clients` parameter only for Linux systems. Use the `-options` parameter for Solaris and AIX systems.

Usage Notes

- You must run this command as `root` user on Linux and UNIX platforms.

Examples

To export with operating system defaults:

```
# srvctl add exportfs -name export1 -id havip1 -path /u01/db1
```

To export read-only to certain clients:

```
# srvctl add exportfs -name export1 -id havip1 -path /u01/db1 -clients node1
-options ro
```

srvctl add filesystem

Adds a device containing a file system (Oracle Automatic Storage Management Cluster File System (Oracle ACFS) or other) to the Oracle Clusterware stack for automount and high availability. This command must be run only one time from the Oracle Grid Infrastructure home.

An Oracle ACFS file system resource is typically created for use with application resource dependency lists. For example, if an Oracle ACFS file system is configured for use as an Oracle Database home, then a resource created for the file system can be included in the resource dependency list of the Oracle Database application. This will cause the file system and stack to be automatically mounted because of the start action of the database application.

Notes:

- Use this command only with Oracle Clusterware.
- To manage Oracle ACFS on Oracle Database 12*c* installations, use the SRVCTL binary in the Oracle Grid Infrastructure home for a cluster (Grid home). If you have Oracle RAC or Oracle Database installed, then you cannot use the SRVCTL binary in the database home to manage Oracle ACFS.

Syntax and Parameters

Use the `srvctl add filesystem` command with the following syntax:

```
srvctl add filesystem -device volume_device -path mountpoint_path
   [-volume volume_name] [-diskgroup disk_group_name]
   [-node node_list | -serverpool server_pool_list] [-user user_list]
   [-fstype {ACFS | EXT3 | EXT4}] [-fsoptions options] [-description description]
   [-appid application_id] [-autostart {ALWAYS | NEVER | RESTORE}]
```

Note: Oracle supports the `EXT3` and `EXT4` values for the `-fstype` parameter only for Linux.

Table F–26 srvctl add filesystem Parameters

Parameter	Description
`-device` *`volume_device`*	Specify the path to the file system volume device you want to add.
`-path` *`mountpoint_path`*	Specify the mount point path that is used to set up dependencies on other resources for nested mounts. This must be an absolute path.
`-volume volume_name`	Specify the name of the volume.
`-diskgroup` *`disk_group_name`*	The name of the Oracle ACFS disk group to which you want to add the device.

`-node node_list |`
`-serverpool` *`server_pool_list`*
	Specify a comma-delimited list of nodes on which to mount the file system device. If you specify a list of nodes or a server pool, then SRVCTL creates a single-node file system resource. Without any nodes or server pools, SRVCTL creates a file system resource that runs on every cluster node.
	Specify a comma-delimited list of server pools in which a file system will run. If you specify a list of server pools, then you dynamically limit the number or type of nodes on which a file system will run. Note that this does not count as a clusterwide file system for later usage that may require a clusterwide Oracle ACFS resource, such as HANFS.
	Note: Nodes and server pools are mutually exclusive.

`-user` *`user_list`*	Specify a comma-delimited list of users authorized to mount and unmount the file system. If you specify no users, then only `root` can mount and unmount the file system. If the file system is for Oracle Database software, then specify the name of the Oracle home owner.

Table F–26 (Cont.) srvctl add filesystem Parameters

Parameter	Description
`-fstype {ACFS | EXT3 | EXT4}`	Specify the type of file system to be mounted. The default value is `ACFS`. If the type of file system to be mounted is not `ACFS`, then it must be a single-node file system, and you must specify `-node` or `-serverpool`.
`-fsoptions` *`options`*	Specify options used to mount the file system. The options are file system and operating system dependent.
`-description` *`description`*	Specify a description of the file system. This is stored as an attribute for later viewing by an administrator.
`-appid` *`application_id`*	Specify an application ID, which is a unique identifier used to group file systems into logical groupings. This method is used to group file systems that may be different on different nodes but are used by an application that must have a dependency on the resource type, which is `ora.id.fs.type`.
`-autostart {ALWAYS | NEVER | RESTORE}`	Specify the file system resource autostart policy. `ALWAYS`: The file system resource always automatically starts. `NEVER`: The file system does not automatically start. `RESTORE`: The file system is restored to its last state. This is the default option.

Usage Notes

You must run this command as `root` user on Linux and UNIX platforms, or as an Administrator user on Windows platforms.

Examples

To add the `d1volume1-295` disk device in the disk group `RAC_DATA` as the Oracle ACFS volume `VOLUME1` with a mount point of `/oracle/cluster1/acfs`:

```
# srvctl add filesystem -device /dev/asm/d1volume1-295
 -path /oracle/cluster1/acfs1
```

To add an Oracle ACFS file system on the dynamic volume device `asm-test-55`, with this file system mounted on one of the specified nodes at a time:

```
# srvctl add filesystem -fstype ACFS -device asm-test-55 -path myacfs
 -node node1,node2,node3
```

srvctl add gns

Use this command to add the Grid Naming Service (GNS) to a cluster when you are using a DHCP public network or to create a client cluster.

Note: Use this command only with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl add gns` command with one of the following syntax models:

To add GNS to a cluster that is not currently running GNS:

```
srvctl add gns [-domain domain_name] -vip {vip_name | ip_address} [-verbose]
```

To change a cluster that is not running GNS to be a client cluster of another cluster that is running GNS:

```
srvctl add gns -clientdata file_name
```

Table F–27 srvctl add gns Parameters

Parameter	Description
`-domain domain_name`	The network subdomain that is used for Forward DNS Lookup of cluster IP addresses. You can only use this parameter if you specify `-vip`, and you must use this parameter if the cluster to which you are adding GNS has a GNS zone delegation and you are using DHCP. The cluster to which you add GNS using `-vip` with the `-domain` parameters becomes a server cluster. If you do not specify `-domain`, then SRVCTL adds GNS without a domain.
`-vip {vip_name |ip_address}`	Specify either the virtual IP (VIP) name or IP address on which GNS listens for DNS requests. You *cannot* use `-vip` with `-clientdata`. Use the `-domain` parameter with `-vip` to make the cluster to which you are adding GNS a server cluster.
`-clientdata path_to_file`	Specify the path to the file you created with the `srvctl export gns` command that contains the GNS credentials. You must copy this file to a node in the cluster you are adding before running the `srvctl add gns` command. The cluster to which you add GNS using `-clientdata` becomes a client cluster. You *cannot* use the `-clientdata` parameter with the `-vip` parameter.
`-verbose`	Verbose output

Usage Notes

- You must run this command as `root` user on Linux and UNIX platforms, or as an Administrator user on Windows platforms.
- When you are adding GNS to a cluster for the first time, use the `-vip` parameter with the `-domain` parameter.
- When you are adding GNS to a cluster using the `-clientdata` parameter, you must first export the GNS data to a file and manually copy the file to a node in the cluster you are adding. Use the `-clientdata` parameter and run this command on any node in the cluster.
- You cannot specify the `-vip` and `-clientdata` parameters at the same time.

Examples

To add GNS to a cluster, making it a server cluster:

```
# srvctl add gns -vip 192.168.16.17 -domain cluster.mycompany.com
```

To add GNS to a cluster, making it a client cluster:

```
# srvctl add gns -clientdata /tmp/gnsdata
```

srvctl add mgmtdb

Adds a management database (CHM repository) resource to the cluster.

Note: Use this command only with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl add mgmtdb` command with the following syntax:

```
srvctl add mgmtdb [-domain domain
```

Table F–28 srvctl add mgmtdb Parameters

Parameter	Description
`-domain domain`	The domain for the database. If you have the `DB_DOMAIN` database initialization parameter set, then you must provide a value for this parameter.

Example

To add a management database to the cluster:

```
$ srvctl add mgmtdb -domain example.com
```

srvctl add mgmtlsnr

Adds a management listener resource (for Cluster Health Monitor) to the cluster.

Note: Use this command only with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl add mgmtlsnr` command with the following syntax:

```
srvctl add mgmtlsnr [-endpoints "[TCP:]port_list[/IPC:key][/NMP:pipe_name
 [/TCPS:s_port][/SDP:port]"] [-skip]]
```

Table F–29 srvctl add mgmtlsnr Parameters

Parameter	Description
`-endpoints "[TCP:]port_list [/IPC:key][/NMP:pipe_name [/TCPS:s_port][/SDP:port]]`	Protocol specifications for the listener. `port_list` is a comma-delimited list of TCP ports or listener endpoints.
`-skip`	Indicates you want to skip the checking of ports.

Example

The following command adds a management listener that is listening on port 1341 to the cluster:

```
$ srvctl add mgmtlsnr -endpoints "TCP:1341"
```

srvctl add mountfs

Adds Network Attached Storage (NAS) configuration to Oracle Clusterware.

Note: Use this command only with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl add mountfs` command with the following syntax:

```
srvctl add mountfs -name mountfs_name -path mount_path -exportserver server_name
   -exportpath path [-mountoptions mount_options] [-user user]
```

Table F–30 srvctl add mountfs Command Parameters

Parameter	Description
`-name` *mountfs_name*	Specify a unique name for the NAS.
`-path` *mount_path*	Specify a mount path for the NAS.
`-exportserver` *server_name*	Specify a name for the export server.
`-exportpath` *path*	Specify an export file path.
`-mountoptions` *mount_options*	Optionally, you can specify mount options for the NAS.
`-user` *user_name*	Optionally, you can specify a user authorized to mount and dismount the NAS.

srvctl add rhpclient

Use this command to add a Rapid Home Provisioning Client configuration file to the Oracle Clusterware configuration.

Syntax and Parameters

Use the `srvctl add rhpclient` command with the following syntax:

```
srvctl add rhpclient -clientdata path_to_file
 [-diskgroup disk_group_list -storage base_path]
```

Notes:

- This command is only available with Oracle Clusterware.
- This command does not create an Oracle Clusterware resource.

Table F–31 srvctl add rhpclient Parameters

Parameter	Description
`-clientdata` *path_to_file*	Specify the path to the file that contains the Rapid Home Provisioning Client data
`-diskgroup` *disk_group_list*	Specify a list of disk groups to create the Oracle ACFS file system for storing images.
`-storage` *base_path*	Specify a location that is available on all cluster nodes and that is used for mounting an Oracle ACFS that you create for image storage on the Rapid Home Provisioning Client.

Usage Notes

- You cannot run this command on a Rapid Home Provisioning Server.

- You must be `root` user on Linux and UNIX platforms, or as an Administrator user on Windows platforms, to run this command.
- You must have a configured Oracle ACFS if you specify the `-storage` parameter.
- The default user is the user that installed Oracle Clusterware, and this user gets added to the user list.

Example

An example of this command is:

```
# srvctl add rhpclient -clientdata /tmp/gnsdata -storage -diskgroups
```

srvctl add rhpserver

Use this command to add a Rapid Home Provisioning Server to the Oracle Clusterware configuration.

Syntax and Parameters

Use the `srvctl add rhpserver` command with the following syntax:

```
srvctl add rhpserver -storage base_path -diskgroup disk_group_name
```

> **Note:** This command is only available with Oracle Clusterware.

Table F–32 srvctl add rhpserver Parameters

Parameter	Description
`-storage` *`base_path`*	Specify a location that is available on all cluster nodes. It is not necessary that this location be shared. The catalog for the server is kept in this location and all the image file systems are mounted on *`base_path`*`/images`.
`-diskgroup` *`disk_group_list`*	Specify a particular disk group from which to create the Oracle ACFS file system for storing images.

Usage Notes

- You must be `root` user on Linux and UNIX platforms, or as an Administrator user on Windows platforms, to run this command.
- The default user is the user that installed Oracle Clusterware, and this user gets added to the user list for the Rapid Home Provisioning Server resource.

srvctl config asm

Displays the configuration for all Oracle ASM instances.

> **Notes:**
>
> - Use this command only with Oracle Clusterware.
> - To manage Oracle ASM on Oracle Database 12*c* installations, use the SRVCTL binary in the Oracle Grid Infrastructure home for a cluster (Grid home). If you have Oracle RAC or Oracle Database installed, then you cannot use the SRVCTL binary in the database home to manage Oracle ASM.

Syntax and Parameters

Use the `srvctl config asm` command with the following syntax:

```
srvctl config asm [-proxy] [-detail]
```

Table F–33 srvctl config asm Parameters

Parameter	Description
`-proxy`	Specifies whether the Oracle ASM configuration to display is a proxy instance.
`-detail`	Print detailed configuration information

Example

The `srvctl config asm` command returns output similar to the following:

```
$ srvctl config asm
ASM Home: /u01/app/12.1.0/grid
ASM Listener: LISTENER
ASM instance count: 3
Password file:
```

The `ASM instance count:` line displays only when Oracle Flex ASM is enabled.

srvctl config exportfs

Displays the configuration for an export file system in Oracle Clusterware.

> **Note:** Use this command only with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl config exportfs` command with the following syntax:

```
srvctl config exportfs [-name unique_name | -id havip_name]
```

Table F–34 srvctl config exportfs Parameters

Parameter	Description
`-name` *unique_name*	Enter the unique name of the NFS export for which you want to display the configuration.
`-id` *havip_name*	Enter the name of a HAVIP to display the configuration of all the NFS exports attached to the specific HAVIP.

Example

If you specify the `-name` parameter, then this command returns output similar to the following, whether you specify `-name` or `-id`:

```
$ srvctl config exportfs -id havip1

export file system kep1 is configured
Exported path: /scratch/ghnew/base/mymount3
Export options:
Exported clients:
export file system kexp2 is configured
Exported path: /scratch/ghnew/base/mymount3
Export options:
```

```
Exported clients:
```

srvctl config filesystem

Displays the configuration for a specific file system resource.

Note: Use this command only with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl config filesystem` command with the following syntax:

```
srvctl config filesystem -device volume_device
```

Use the `-device` parameter to specify the path to the file system resource for which you want to display the configuration.

Examples

The following example lists the configuration of all file systems:

```
$ srvctl config filesystem
```

The following example displays the configuration for a specific device:

```
srvctl config filesystem -device /dev/asm/d1volume1-295
```

srvctl config gns

Displays the configuration for GNS.

Note: Use this command only with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl config gns` command with the following syntax:

```
srvctl config gns [-subdomain] [-multicastport] [-node node_name]
  [-port] [-status] [-version] [-query name] [-list] [-clusterguid]
  [-clustername] [-clustertype] [-loglevel] [-network] [-detail]
```

Table F–35 srvctl config gns Parameters

Parameter	Description
`-subdomain`	Display the subdomain served by GNS.
`-multicastport`	Display the port on which the GNS daemon is listening for multicast requests.
`-node` *`node_name`*	Display the configuration information for GNS on the specified node.
`-port`	Display the port that the GNS daemon uses to communicate with the DNS server.
`-status`	Display the status of GNS.
`-version`	Display the version of GNS.
`-query` *`name`*	Query GNS for the records belonging to a specific name.
`-list`	List all records in GNS.

Table F–35 (Cont.) srvctl config gns Parameters

Parameter	Description
`-clusterguid`	Display the globally unique identifier of the cluster where GNS is running.
`-clustername`	Display the name of the cluster where GNS is running.
`-clustertype`	Display the type of configuration of GNS on this cluster.
`-loglevel`	Print the log level of the GNS.
`-network`	Display network on which GNS is listening.
`-detail`	Print detailed configuration information about the GNS.

srvctl config mgmtdb

Displays configuration information for the management database (CHM repository) resource.

Note: Use this command only with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl config mgmtdb` command with the following syntax:

```
srvctl config mgmtdb [-verbose] [-all]
```

Table F–36 srvctl config mgmtdb Parameters

Parameter	Description
`-verbose`	Display verbose output
`-all`	Print detailed configuration information

srvctl config mgmtlsnr

Displays configuration information for the management listener resource (for CHM).

Note: Use this command only with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl config mgmtlsnr` command with the following syntax:

```
srvctl config mgmtlsnr [-all]
```

Use the `-all` parameter to print detailed configuration information.

srvctl config mountfs

Displays the configuration information for a specific NAS.

Note: Use this command only with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl config mountfs` command with the following syntax:

```
srvctl config mountfs [-name mountfs_name]
```

Specify the unique name of the NAS for which you want to display configuration information.

srvctl config rhpclient

Use this command to display configuration information for a Rapid Home Provisioning Client instance.

Syntax and Parameters

Use the `srvctl config rhpclient` command with the following syntax:

```
srvctl config rhpclient
```

Note: This command is only available with Oracle Clusterware.

Example

This command returns output similar to the following:

```
$ srvctl config rhpclient

Rapid Home Provisioning Client is configured
Cluster name: mjk9394
Storage base path: /scratch/aime/storage
Disk group: MJKRHPCDG
Rapid Home Provisioning Server (RHPS): mjk8990clust
Rapid Home Provisioning Server discovery string: 198.51.100.191
Port number: 23795
Rapid Home Provisioning Client is enabled
Rapid Home Provisioning Client is individually enabled on nodes:
Rapid Home Provisioning Client is individually disabled on nodes:
```

srvctl config rhpserver

Use this command to display configuration information for a Rapid Home Provisioning Server.

Syntax and Parameters

Use the `srvctl config rhpserver` command with the following syntax:

```
srvctl config rhpserver
```

Note: This command is only available with Oracle Clusterware.

Example

This command returns output similar to the following:

```
$ srvctl config rhpserver

Storage base path: /scratch/aime/storagedg
Disk Groups: MJKRHPSDG
Port number: 23795
Rapid Home Provisioning Server is enabled
Rapid Home Provisioning Server is individually enabled on nodes:
```

```
Rapid Home Provisioning Server is individually disabled on nodes:
```

srvctl disable asm

Disables the Oracle ASM proxy resource. Oracle ASM will restart if it contains Oracle Clusterware data or if the node restarts and it was running before the node failed. The `srvctl disable asm` command also prevents the `srvctl start asm` command from starting Oracle ASM instances.

Disabling the Oracle ASM proxy resource prevents the databases and disk groups from starting because they depend on the Oracle ASM proxy resource.

Notes:

- To manage Oracle ASM on Oracle Database 12*c* installations, use the SRVCTL binary in the Oracle Grid Infrastructure home for a cluster (Grid home). If you have Oracle RAC or Oracle Database installed, then you cannot use the SRVCTL binary in the database home to manage Oracle ASM.
- Oracle ASM is part of the Oracle Clusterware stack and when OCR and voting files are stored on Oracle ASM, then Oracle ASM starts when OHASD starts the Oracle Clusterware stack. The `srvctl disable asm` command does not prevent the Oracle ASM instance managed by OHASD from starting. Oracle ASM, therefore, starts as needed by the Oracle Clusterware stack.

  The `srvctl disable asm` command prevents the Oracle ASM Oracle Clusterware proxy resource, and any resources that depend on it, from starting. So, the command prevents Oracle Clusterware-managed objects, such as databases, disk groups, and file systems that depend on Oracle ASM, from starting. For example, the `srvctl start database | diskgroup | filesystem` command fails to start any of those objects on nodes where the Oracle ASM Oracle Clusterware proxy resource is disabled. The command also prevents the `srvctl start asm` command from starting Oracle ASM on remote nodes.

Syntax and Parameters

Use the `srvctl disable asm` command with the following syntax:

```
srvctl disable asm [-proxy] [-node node_name]
```

Table F–37 srvctl disable asm Parameters

Parameter	Description
`-proxy`	Specifies whether you are disabling a proxy Oracle ASM configuration.
`-node` *`node_name`*	Node name **Note:** Use this parameter only with Oracle Clusterware.

Example

The following example disables Oracle ASM on a node named `crmnode1`:

```
$ srvctl disable asm -node crmnode1
```

srvctl disable exportfs

Disables an export file system managed by Oracle Clusterware.

> **Note:** Use this command only with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl disable exportfs` command with the following syntax:

```
srvctl disable exportfs -name filesystem_name
```

Table F–38 srvctl disable exportfs Parameters

Parameter	Description
`-name` *`filesystem_name`*	Enter a unique name for the NFS export file system you want to disable. This parameter is required.

Usage Notes

- You must run this command as `root` user on Linux and UNIX platforms.

Example

An example of this command is:

```
# srvctl disable exportfs -name export1
```

srvctl disable mountfs

Disables the specified NAS from Oracle Clusterware management.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl disable mountfs` command with the following syntax:

```
srvctl disable mountfs -name mountfs_name [-node node_list]
```

Table F–39 srvctl disable mountfs

Parameter	Description
`-name` *`mountfs_name`*	Specify the unique name for the NAS you want to disable.
`-node` *`node_list`*	Optionally, you can specify a comma-delimited list of nodes on which to disable NAS.

srvctl disable rhpclient

Use this command to disable a Rapid Home Provisioning Client instance on a specific node from Oracle Clusterware management.

Syntax and Parameters

Use the `srvctl disable rhpclient` command with the following syntax:

```
srvctl disable rhpclient [-node node_name]
```

Specify the node on which you want to disable the Rapid Home Provisioning Client instance.

Note: This command is only available with Oracle Clusterware.

Table F–40 srvctl disable rhpclient Parameters

Parameter	Description
`-node` *`node_name`*	Name of a node in the cluster

Example

An example of this command is:

```
srvctl disable rhpclient -node clusterNode01
```

srvctl disable rhpserver

Use this command to disable a Rapid Home Provisioning Server on a specific node from Oracle Clusterware management.

Syntax and Parameters

Use the `srvctl disable rhpserver` command with the following syntax:

```
srvctl disable rhpserver [-node node_name]
```

Specify the node on which you want to disable the Rapid Home Provisioning Server.

Note: This command is only available with Oracle Clusterware.

Table F–41 srvctl disable rhpserver Parameters

Parameter	Description
`-node` *`node_name`*	Name of a node in the cluster

Example

An example of this command is:

```
srvctl disable rhpserver -node clusterNode01
```

srvctl enable exportfs

Enables an export file system configuration in Oracle Clusterware.

Note: This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl enable exportfs` command with the following syntax:

```
srvctl enable exportfs -name filesystem_name
```

Table F–42 srvctl enable exportfs Parameters

Parameter	Description
`-name` *`filesystem_name`*	Enter a unique name for the NFS export file system you want to enable. This parameter is required.

Usage Notes

- You must run this command as `root` user on Linux and UNIX platforms.

Example

An example of this command is:

```
# srvctl enable exportfs -name export1
```

srvctl enable mountfs

Enables the specified Network Attached Storage (NAS) for Oracle Clusterware management.

Note: This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl enable mountfs` command with the following syntax:

```
srvctl enable mountfs -name mountfs_name [-node node_list]
```

Table F–43 srvctl enable mountfs

Parameter	Description
`-name` *`mountfs_name`*	Specify the unique name for the NAS you want to enable.
`-node` *`node_list`*	Optionally, you can specify a comma-delimited list of nodes on which to enable NAS.

srvctl enable rhpclient

Use this command to enable a Rapid Home Provisioning Client instance on a specific node.

Syntax and Parameters

Use the `srvctl enable rhpclient` command with the following syntax:

```
srvctl enable rhpclient [-node node_name]
```

You can optionally specify the node on which you want to enable the Rapid Home Provisioning Client instance. If you do not specify a node, then SRVCTL enables all of the Rapid Home Provisioning Clients in the Oracle Clusterware configuration.

Note: This command is only available with Oracle Clusterware.

Example

An example of this command is:

```
srvctl enable rhpclient -node clusterNode03
```

srvctl enable rhpserver

Use this command to enable a Rapid Home Provisioning Server on a specific node.

Syntax and Parameters

Use the `srvctl enable rhpserver` command with the following syntax:

```
srvctl enable rhpserver [-node node_name]
```

Specify the node on which you want to enable the Rapid Home Provisioning Server. If you do not specify a node, then SRVCTL enables all of the Rapid Home Provisioning Servers in the Oracle Clusterware configuration.

> **Note:** This command is only available with Oracle Clusterware.

Example

An example of this command is:

```
srvctl enable rhpserver -node clusterNode03
```

srvctl modify exportfs

Modifies an export file system configuration in Oracle Clusterware.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl modify exportfs` command with the following syntax:

```
srvctl modify exportfs -name unique_name [-path path_to_export]
    [-clients node_list] [-options nfs_options_string]
```

Table F–44 srvctl modify exportfs Parameters

Parameter	Description
`-name` *`unique_name`*	Enter the unique name of the NFS export file system to modify. This parameter is required.
`-path` *`path_to_export`*	You can modify the path to the Oracle Automatic Storage Management Cluster File System (Oracle ACFS) where the export is located.

`-clients` *`node_list`*	You can modify the comma-delimited list of clients (hosts) to where the file system is exported. You can enter net groups and wildcards. For example:
	`-clients @netgroup1,`
	`*.us.domain.com,`
	`16.140.0.0, hostname1`
	Note: You can use the `-clients` parameter only for Linux systems. Use the `-options` parameter for Solaris and AIX systems.

Table F–44 (Cont.) srvctl modify exportfs Parameters

Parameter	Description
`-options` *nfs_client_string*	Use this parameter to modify the export options for the exportfs file system.

Usage Notes

- You must run this command as `root` user on Linux and UNIX platforms.
- You must start and stop the NFS export before any changes you make take effect.

Example

An example of this command is:

```
# srvctl modify exportfs -name exportfs1 -path /mnt/racdb1
```

srvctl modify mountfs

Modifies Network Attached Storage (NAS) configuration in Oracle Clusterware.

> **Note:** This command is available only with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl modify mountfs` command with the following syntax:

```
srvctl modify mountfs -name mountfs_name [-path mount_path] [-exportserver server_
name]
   [-exportpath path] [-mountoptions mount_options] [-user user]
```

Table F–45 srvctl modify mountfs Command Parameters

Parameter	Description
`-name` *mountfs_name*	Specify a unique name for the NAS you want to modify.
`-path` *mount_path*	Modify the mount path for the NAS.
`-exportserver` *server_name*	Modify the name of the export server.
`-exportpath` *path*	Modify the export file path.
`-mountoptions` *mount_options*	Modify mount options for the NAS.
`-user` *user_name*	Modify the user authorized to mount and dismount the NAS.

srvctl modify rhpclient

Use this command to change the locations of the client data and image storage for the Rapid Home Provisioning Client.

Syntax and Parameters

Use the `srvctl modify rhpclient` command with the following syntax:

```
srvctl modify rhpclient [-clientdata path_to_client_data] [-port rmi_port]
   [-force]
```

Note: This command is only available with Oracle Clusterware.

Table F–46 srvctl modify rhpclient Parameters

Parameter	Description
`-clientdata` *`path_to_client_data`*	Specify the path to the Rapid Home Provisioning Client data
`-port` *`rmi_port`*	Modifies the RMI port number used by the Rapid Home Provisioning Client
`-force`	Force this command to proceed

Usage Notes

- You must run this command as the `root` user on Linux and UNIX platforms, or as an Administrator user on Windows platforms

srvctl modify rhpserver

Use this command to modify the storage for a Rapid Home Provisioning Server.

Syntax and Parameters

Use the `srvctl modify rhpserver` command with the following syntax:

```
srvctl modify rhpserver -storage base_path [-port rmi_port] [-force]
```

Note: This command is only available with Oracle Clusterware.

Table F–47 srvctl modify rhpserver Parameters

Parameter	Description
`-storage` *`base_path`*	Specify the new location that is available on all cluster nodes and that is used for mounting an Oracle ACFS that you create for image storage on the Rapid Home Provisioning Server
`-port` *`rmi_port`*	Modifies the RMI port number used by the Rapid Home Provisioning Server
`-force`	This option stops and restarts the resource to effect a change

srvctl relocate rhpclient

Use this command to relocate a Rapid Home Provisioning Client instance to a different node.

Syntax and Parameters

Use the `srvctl relocate rhpclient` command with the following syntax:

```
srvctl relocate rhpclient -node node_name
```

Specify a node to which you want to relocate the Rapid Home Provisioning Client instance.

Note: This command is only available with Oracle Clusterware.

Example

An example of this command is:

```
srvctl relocate rhpclient -node crsNode03
```

srvctl relocate rhpserver

Use this command to relocate a Rapid Home Provisioning Server to a different node.

Syntax and Parameters

Use the `srvctl relocate rhpserver` command with the following syntax:

```
srvctl relocate rhpserver [-node node_name]
```

Specify a node to which you want to relocate the Rapid Home Provisioning Server.

Note: This command is only available with Oracle Clusterware.

Example

An example of this command is:

```
srvctl relocate rhpserver -node crsNode03
```

srvctl remove exportfs

Removes the specified export file system configuration.

Note: This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl remove exportfs` command with the following syntax:

```
srvctl remove exportfs -name exportfs_name [-force]
```

Table F–48 srvctl remove exportfs Parameters

Parameter	Description
`-name` *`exportfs_name`*	Enter the unique name of the NFS export configuration you want to remove.
`-force`	Forcibly remove the resource, ignoring all dependencies.

Usage Notes

- You must run this command as `root` user on Linux and UNIX platforms.
- If you do not use the `-force` parameter, then you must first stop the NFS export you want to remove before you remove it.
- If you use the `-force` parameter, the resulting configuration can cause errors when attempting to shut down the Oracle Clusterware stack.

Examples

To remove the stopped export file system named `export1`:

```
# srvctl remove exportfs -name export1
```

To remove a running export, leaving it exported:

```
# srvctl remove exportfs -name export1 -force
```

srvctl remove mountfs

Removes a specific NAS configuration.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl remove network command as follows:

```
srvctl remove mountfs -name mountfs_name [-force]
```

Table F–49 srvctl remove mountfs

Parameter	Description
`-name` *`mountfs_name`*	Specify the unique name for the NAS to mount.
`-force`	Specify `-force` to ignore resource dependencies while removing the NAS.

srvctl remove rhpclient

Use this command to remove a Rapid Home Provisioning Client instance.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl remove rhpclient` command with the following syntax:

```
srvctl remove rhpclient [-force]
```

Use the `-force` parameter to forcibly remove the Rapid Home Provisioning Client instance, ignoring any dependencies.

srvctl remove rhpserver

Use this command to remove a Rapid Home Provisioning Server instance configured for the cluster.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl remove rhpserver` command with the following syntax:

```
srvctl remove rhpserver [-force]
```

Use the `-force` parameter to forcibly remove the Rapid Home Provisioning Server, ignoring any dependencies.

srvctl start exportfs

Starts an export file system configuration in Oracle Clusterware.

Note: This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl start exportfs` command with the following syntax:

```
srvctl start exportfs {-name unique_name | -id havip_name}
```

Table F–50 srvctl start exportfs Parameters

Parameter	Description
`-name` *`unique_name`*	Enter the unique name of the NFS export you want to start.
`-id` *`havip_name`*	The unique ID associated with the HAVIP resource.

Example

To start an export file system:

```
$ srvctl start exportfs -name export1
```

srvctl start mountfs

Mounts a specific NAS.

Note: This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl start mountfs` command with the following syntax:

```
srvctl start mountfs -name mountfs_name [-node node_list]
```

Table F–51 srvctl start mountfs

Parameter	Description
`-name` *`mountfs_name`*	Specify the unique name for the NAS to mount.
`-node` *`node_list`*	Optionally, you can specify a comma-delimited list of nodes on which to mount NAS.

srvctl start rhpclient

Use this command to start a Rapid Home Provisioning Client instance on one or more nodes.

Note: This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl start rhpclient` command with the following syntax:

```
srvctl start rhpclient [-node node_name]
```

Specify the node on which you want to start the Rapid Home Provisioning Client instance. If you do not specify a node, then Oracle Clusterware determines the node where the Rapid Home Provisioning Client is to be started. The Rapid Home Provisioning Client resource has cardinality of 1 and Oracle Clusterware decides the placement when the Rapid Home Provisioning Client resource is started.

Example

An example of this command to start the Rapid Home Provisioning Server on the node named `crs3`:

```
$ srvctl start rhpclient -node crs3
```

srvctl start rhpserver

Use this command to start a Rapid Home Provisioning Server on a specific node.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl start rhpserver` command with the following syntax:

```
srvctl start rhpserver [-node node_name]
```

Specify the node on which you want to start the Rapid Home Provisioning Server.

Example

For example, to start the Rapid Home Provisioning Server on the node named `clusterNode03`:

```
srvctl start rhpserver -node clusterNode03
```

srvctl status exportfs

Displays the status of an export file system configuration.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl status exportfs` command with the following syntax:

```
srvctl status exportfs [-name unique_name | -id havip_name]
```

Table F–52 srvctl status exportfs Parameters

Parameter	Description
`-name` *`unique_name`*	Enter the unique name of the NFS export for which you want to display the status.
`-id` *`havip_name`*	Optionally, you can specify the name of an HAVIP resource and display the status of all of the NFS exports associated with the specified HAVIP resource.

Usage Notes

If you do not specify any parameters, then SRVCTL displays the status for all NFS exports in the cluster.

Example

This command returns output similar to the following:

```
$ srvctl status exportfs

export file system export1 is enabled
export file system export1 is not exported
export file system export2 is enabled
export file system export2 is exported on node node1
```

srvctl status mountfs

Displays the current state of a specific NAS.

Note: This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl status mountfs` command with the following syntax:

```
srvctl status mountfs -name mountfs_name
```

Specify the unique name of the NAS for which you want to display the status.

srvctl status rhpclient

Displays the current state of a Rapid Home Provisioning Client instance.

Note: This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl status rhpclient` command with the following syntax:

```
srvctl status rhpclient
```

This command does not accept any parameters, other than `-help`.

Example

This command displays output similar to the following:

```
$ srvctl status rhpclient

Rapid Home Provisioning Client is enabled
Rapid Home Provisioning Client is running on node mjk1270093
```

srvctl status rhpserver

Displays the current state of a Rapid Home Provisioning Server.

Note: This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl status rhpserver` command with the following syntax:

```
srvctl status rhpserver
```

This command does not accept any parameters, other than `-help`.

Example

This command displays output similar to the following:

```
$ srvctl status rhpserver

Rapid Home Provisioning Server is enabled
Rapid Home Provisioning Server is running on node mjk1270089
```

srvctl stop exportfs

Use this command to stop an export file system configuration managed by Oracle Clusterware.

Note: This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl stop exportfs` command with the following syntax:

```
srvctl stop exportfs {-name unique_name | -id havip_name} [-force]
```

Table F–53 srvctl stop exportfs Parameters

Parameter	Description
`-name` *`unique_name`*	Enter the unique name of the NFS export to stop.
`-id` *`havip_name`*	If you specify this parameter, then SRVCTL stops all of the NFS exports associated with the HAVIP resource you specify.
`-force`	Stop the NFS export, ignoring errors

Usage Notes

- You must run this command as `root` user on Linux and UNIX platforms.

Example

An example of this command is:

```
# srvctl stop exportfs -name export1
```

srvctl stop mountfs

Unmounts a specific NAS.

Note: This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl stop mountfs` command with the following syntax:

```
srvctl stop mountfs -name mountfs_name [-node node_list] [-force]
```

Table F–54 srvctl stop mountfs

Parameter	Description
-name *mountfs_name*	Specify the unique name for the NAS to unmount.
-node *node_list*	Optionally, you can specify a comma-delimited list of nodes from which to unmount NAS.
-force	Specify -force to ignore resource dependencies while stopping NAS.

srvctl stop rhpclient

Use this command to stop a Rapid Home Provisioning Client instance that is in a running or starting state.

Note: This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl stop rhpclient command with the following syntax:

```
srvctl stop rhpclient
```

This command does not accept any parameter, except for the -help parameter.

Example

An example of this command is:

```
$ srvctl stop rhpclient
```

srvctl stop rhpserver

Use this command to stop a Rapid Home Provisioning Server that is in a running or starting state.

Note: This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl stop rhpserver command with the following syntax:

```
srvctl stop rhpserver
```

This command does not accept any parameter, except for the -help parameter.

Example

An example of this command is:

```
srvctl stop rhpserver
```

G

Oracle Clusterware Agent Framework C Application Program Interfaces

Oracle Clusterware manages applications registered as resources with Oracle Clusterware and must have access to application-specific primitives that have the ability to start, stop, and monitor a specific resource. Oracle Clusterware runs all resource-specific commands through an entity called an *agent*.

An agent is a program that contains the agent framework and user code to manage resources. The agent framework is a C library that enables you to plug in your application-specific code to manage customized applications. You program all of the actual application management functions in C or C++, such as starting, stopping, and checking the health of an application, into the agent. These functions are referred to as *entry points*. The agent framework is responsible for invoking these entry point functions on behalf of Oracle Clusterware. Agent developers can use these entry points to plug in the required functionality for a specific resource regarding how to start, stop, and monitor a resource. Agents are capable of managing multiple resources.

This appendix includes the following topics:

- Agent Framework Data Types
- Agent Framework Context Initialization and Persistence
- Prototype for C and C++ Entry Point Functions
- C and C++ Entry Point Types and Codes
- C and C++ Entry Point Function Return Values
- Multithreading Considerations
- Deprecated APIs
- API Reference
- Agent Example

Agent Framework Data Types

The Oracle Clusterware agent framework uses Oracle-defined portable data types to represent strings and numbers. By using these data types, you can build and execute agent programs on multiple platforms. Table G–1 lists and describes the Oracle data types used in the agent framework.

Table G–1 Oracle Data Types Used in the Agent Framework

Oracle Data Type	Description
ub4	Unsigned 32-bit integer
sb4	Signed 32-bit integer
oratext*	C style null-terminated strings, explicit cast to **char** * required

The agent framework APIs return a status value of type `clsagfwret`, as follows:

`CLSAGFW_SUCCESS`: Function executed successfully
`CLSAGFW_FAIL`: There was error in the parameters or in execution of the function

String values returned by an API are only accessible in the scope of the current command entry point. Agents can copy the returned string value to its own buffers to retain the value.

Agent Framework Context Initialization and Persistence

The agent framework is initialized by the `clsagfw_init()` function call. There is no context that you need to maintain. Action entry point functions are passed a pointer to the agent framework context (type `clsagfw_aectx`). This context is specific to the entry point being called and cannot be used beyond the scope of the entry point. Other agent framework APIs take this context pointer as a parameter. This is an opaque data structure that the agent framework maintains but does not explicitly initialize or destroy.

Prototype for C and C++ Entry Point Functions

The agent implements the entry point functions for the resource management commands and registers these functions with the agent framework. The agent framework calls an entry point function to execute a command or action on a resource. The entry point functions need to conform to the following prototype and convention:

```
typedef ub4 (*clsagfw_action_entry) (const clsagfw_aectx *aectx)
```

The entry point function is called with a pointer to a context area that the agent framework maintains. This function should return a value indicating the status of execution of the entry point. For the CHECK entry point, the function returns the current state of the resource.

If you develop the agent using C++, then you declare the entry point function prototypes using the **`extern 'C'`** keyword.

C and C++ Entry Point Types and Codes

The agent framework currently defines seven entry point types corresponding to resource management commands and operations performed on a resource. Agents must register entry points for START, STOP, CHECK, and CLEAN commands. Other command entry points are optional. The agent registers entry points by calling the `clsagfw_set_entrypoint()` API for each resource type and command. Table G–2 lists and describes entry point types and their equivalent type codes.

Table G–2 Entry Point Types

Name	Description	Type Code
START	Start a resource and bring it online	CLSAGFW_ACTION_START
STOP	Stop a resource and take it offline	CLSAGFW_ACTION_STOP
CLEAN	Clean up after a failed command on the resource	CLSAGFW_ACTION_CLEAN
CHECK	Check the state of the resource	CLSAGFW_ACTION_CHECK
ABORT	End an executing command that has timed out	CLSAGFW_ACTION_ABORT
DELETE	Resource is about to be deleted	CLSAGFW_ACTION_RES_DELETED
MODIFY	Resource is modified	CLSAGFW_RESATTR_MODIFIED
ACTION	Invoke custom action on the resource	CLSAGFW_RES_ACTION

> **See Also:** "Oracle Clusterware Resources and Agents" on page 9-1 for more information about entry points

C and C++ Entry Point Function Return Values

All entry point functions, except for the CHECK entry point, return one of the following values to indicate the status of execution of the entry point:

- **CLSAGFW_AE_SUCCESS**: Entry point execution completed successfully
- **CLSAGFW_AE_FAIL**: Entry point execution failed

The agent framework calls the CHECK entry point function to monitor the state of the resource. This function returns one of the following values that represents the current state of the resource:

CLSAGFW_ONLINE
CLSAGFW_UNPLANNED_OFFLINE
CLSAGFW_PLANNED_OFFLINE
CLSAGFW_UNKNOWN
CLSAGFW_PARTIAL
CLSAGFW_FAILED

> **See Also:** "Agents" on page 9-5 for more information about evaluating and returning the state of a resource

Multithreading Considerations

The Oracle Clusterware agent framework is multithreaded by design. The agent code can also start new threads to execute application-specific functionality, such as monitoring external event sources. The agent framework executes only one entry point of a resource at a time, with the only exception being the ABORT entry point which is invoked to end a currently active action on a resource. C and C++ developers must ensure that all agent code is multithread safe. Access to global variables and other resources must be serialized using locking mechanisms, such mutexes. Other external C or C++ libraries included by the agent should also be multithread safe.

Deprecated APIs

Table G–3 lists the deprecated agent framework APIs and the corresponding replacement APIs for Oracle Clusterware.

Table G–3 Deprecated Agent Framework APIs

Deprecated API	Replacement
`clsagfw_exit()`	`clsagfw_exit2()`
`clsagfw_send_status()`	`clsagfw_send_status2()`

API Reference

This section lists and describes the following functions:

- `clsagfw_add_type()`
- `clsagfw_check_resource()`
- `clsagfw_create_attr_iterator()`
- `clsagfw_delete_cookie()`
- `clsagfw_exit2()`
- `clsagfw_get_attr_from_iterator()`
- `clsagfw_get_attrvalue()`
- `clsagfw_get_check_type()`
- `clsagfw_get_cmdid()`
- `clsagfw_get_cookie()`
- `clsagfw_get_request_action_name()`
- `clsagfw_get_resource_id()`
- `clsagfw_get_resource_name()`
- `clsagfw_get_retry_count()`
- `clsagfw_get_type_name()`
- `clsagfw_init()`
- `clsagfw_is_cmd_timedout()`
- `clsagfw_log()`
- `clsagfw_modify_attribute()`
- `clsagfw_reset_attr_iterator()`
- `clsagfw_send_status2()`
- `clsagfw_set_cookie()`
- `clsagfw_set_entrypoint()`
- `clsagfw_set_exitcb()`
- `clsagfw_set_resource_state_label()`
- `clsagfw_startup()`

clsagfw_add_type()

The clsagfw_add_type() function registers a resource type that the agent manages.

Syntax

```
clsagfwret clsagfw_add_type(const oratext *type_name);
```

Input

type_name: Resource type name that the agent is to manage

Notes

On startup, an agent must register all the resource types that it manages. A single agent can manage multiple resource types and multiple resources.

clsagfw_check_resource()

The clsagfw_check_resource() function initiates a check action on a specific resource.

Syntax

```
void clsagfw_check_resource(const oratext *pResId)
```

Parameters

pResId: Identification of the resource

Notes

Agents can use this function in entry points or in other background threads to initiate a check action on a specific resource. The agent framework schedules and executes the check action in another thread. This function has no effect if invoked from a CHECK entry point.

clsagfw_create_attr_iterator()

The clsagfw_create_attr_iterator() function creates an iterator to access all the attributes of the current resource.

Syntax

```
clsagfwret clsagfw_create_attr_iterator(const clsagfw_aectx *pAeCtx,
                                        clsagfw_ai_flags    flags);
```

Input

pAeCtx: Agent framework context pointer that is passed to entry points

flags: CLSAGFW_ALL_ATTRS: All attributes of the resource

CLSAGFW_MODIFIED_ATTRS: Only modified attributes

Notes

Agents call this function to create an iterator to navigate through all the attributes of the resource. This is useful for agents managing multiple resources of different types. The iterator is an internal iterator maintained in the agent framework context. Each attribute is retrieved by calling the clsagfw_get_attr_from_iterator() API. Modified attributes can be accessed only in the MODIFY entry point. At any given time, an entry point can create only one iterator. All subsequent attempts to create an iterator gets the pointer to the iterator that is already created.

clsagfw_delete_cookie()

The `clsagfw_delete_cookie()` function deletes a cookie stored in the agent framework.

Syntax

```
clsagfwret clsagfw_delete_cookie(const oratext *key);
```

Input

`key`: Key of the cookie, null terminated string

Notes

This function deletes the cookie and frees all the memory used by the cookie and its value.

clsagfw_exit2()

The `clsagfw_exit2()` function terminates the agent process with resource retry feedback.

Syntax

```
void clsagfw_exit2(const oratext *rid, sb4 exitCode, bool retry);
```

Input

`rid`: Resource ID of the resource causing the irrecoverable error

`exitCode`: Process exit code

`retry`: option to retry resource

Notes

An agent can encounter a irrecoverable error when executing an entry point for a resource and then must terminate the current process. An agent can use this API to end the agent process and provide feedback to the agent framework on the resource that is the cause for the irrecoverable error. If the retry parameter is set to *true*, then the current command on the resource is retried when the agent process restarts. If the parameter is set to *false*, then the current resource command is not retried and monitoring of the resource is disabled. This option can be used when a particular resource is the cause of repeated failures of an agent. The Oracle Clusterware server keeps track of the resources causing agent exits and automatically disables a resource if the failure rate is high. To re-enable a resource disabled due to high failure rate, must issue an explicit request to start the resource (for example, using CRSCTL).

The Oracle Clusterware server restarts the agent to continue monitoring resources managed by the agent.

clsagfw_get_attr_from_iterator()

The `clsagfw_get_attr_from_iterator()` function returns the next attribute name and value from the created iterator.

Syntax

```
clsagfwret clsagfw_get_attr_from_iterator(const clsagfw_aectx *pAeCtx,
                                          const oratext **argName,
                                          const oratext **argValue);
```

Input

pAeCtx: Agent framework context pointer that is passed to entry points

Output

argname: Attribute name

argValue: Attribute value

Notes

Use this function to retrieve attributes after the iterator has been created in the agent framework context. The iterator is advanced to the next attribute when this function returns. If the API returns CLSAGFW_FAIL, then there are no more attributes to retrieve for the resource.

clsagfw_get_attrvalue()

The clsagfw_get_attrvalue() function gets the value of an attribute of the current resource.

Syntax

```
void clsagfw_get_attrvalue(const clsagfw_aectx *pAeCtx,
                           const oratext *pAttrName,
                           const oratext **pAttrValue);
```

Input

pAeCtx: Agent framework context pointer that is passed to entry points

pAttrName: Name of attribute to retrieve

Output

pAttrValue: Value of the specified attribute

Notes

Agents use this API to retrieve the value of a single attribute of the current command resource.

clsagfw_get_check_type()

The clsagfw_get_check_type() function returns the type of the check command currently being executed.

Syntax

```
clsagfw_checktype
clsagfw_get_check_type(const clsagfw_aectx *pAeCtx);
```

Input

pAeCtx: Agent framework context pointer that is passed to entry points

Notes

The agent framework invokes the CHECK entry point of a resource for multiple conditions. By using this function, the agent can identify the exact condition for the current CHECK entry point and perform appropriate actions.

Possible conditions include:

- **CLSAGFW_INITIAL_CHECK**: Check entry point invoked as part of the initial probe stage when the Oracle Clusterware server is started or restarted.
- **CLSAGFW_CHECK_AFTER_CMD**: Check entry point invoked to follow up the prior action (such as start or stop) to determine the resultant resource state after the action execution.
- **CLSAGFW_PERIODIC_CHECK**: Check entry point invoked by the agent framework at periodic intervals to monitor the state of a resource. The time interval is specified in the `CHECK_INTERVAL` attribute of the resource.
- **CLSAGFW_EXTERNAL_CHECK**: Check entry point invoked by agent framework when it receives an explicit check request from external entities, such as CRSCTL.
- **CLSAGFW_CHECKFROM_EP**: Check action initiated by the agent itself by calling the `clsagfw_check_resource()` API.

clsagfw_get_cmdid()

The `clsagfw_get_cmdid()` function returns the type of the command for which the entry point is invoked.

Syntax

```
clsagfw_aecode clsagfw_get_cmdid(const clsagfw_aectx *pAeCtx);
```

Input

`pAeCtx`: Agent framework context pointer that is passed to entry points

Notes

The agent can call this API in the ABORT entry point to determine the resource command that is being aborted.

clsagfw_get_cookie()

The `clsagfw_get_cookie()` function retrieves the value of a cookie stored in the agent framework.

Syntax

```
clsagfwret clsagfwret clsagfw_set_cookie(const oratext *key,
                                          const void **value);
```

Input

`key`: Key of the cookie, null terminated string

`Value`: Pointer to the value of the cookie

Notes

None

clsagfw_get_request_action_name()

The `clsagfw_get_request_action_name()` function returns the name of the custom action to be executed on a resource.

Syntax

```
const oratext *
clsagfw_get_request_action_name(const clsagfw_aectx *pAeCtx);
```

Input

`pAeCtx`: Agent framework context that is passed to user entry points

Notes

This function is called in the ACTION entry point to retrieve the action name that you specified in a request action command. You configure the list of actions for a resource in the ACTION attribute of the resource.

clsagfw_get_resource_id()

The `clsagfw_get_resource_id()` function retrieves the ID of the resource for which the entry point is being executed.

Syntax

```
const oratext* clsagfw_get_resource_id(const clsagfw_aectx *pAeCtx);
```

Input

`pAeCtx`: Agent framework context pointer that is passed to entry points

Notes

This function returns the Oracle Clusterware-generated resource ID. To retrieve the public display name of the resource, use the `clsagfw_get_resource_name()` API.

clsagfw_get_resource_name()

The `clsagfw_get_resource_name()` function retrieves the resource name of the current command.

Syntax

```
const oratext*
  clsagfw_get_resource_name(const clsagfw_aectx *pAeCtx);
```

Input

`pAeCtx`: Agent framework context pointer that is passed to entry points

Notes

This function returns the name of the resource for which the current command and entry point are being executed.

clsagfw_get_retry_count()

The `clsagfw_get_retry_count()` function returns the retry attempt number for the current command.

Syntax

```
ub4 clsagfw_get_retry_count(const clsagfw_aectx *pAeCtx)
```

Input

`pAeCtx`: Agent framework context that is passed to user entrypoints

Notes

Agents can call this function in entrypoints to check if the current command is being retried. An agent process or Oracle Clusterware server process can terminate in the middle of executing a command on a resource. Oracle Clusterware retries the failed command when the agent or server process restarts. The command may have successfully or partially executed in the previous attempt. The agent can perform resume or clean up actions when Oracle Clusterware retries the command. The number of times a command can be retried is an internal parameter of Oracle Clusterware.

clsagfw_get_type_name()

The `clsagfw_get_type_name()` function retrieves the type name of the resource for which the entry point is being executed.

Syntax

```
const oratext* clsagfw_get_type_name(const clsagfw_aectx *pAeCtx);
```

Input

`pAeCtx`: Agent framework context pointer that is passed to entry points

Notes

This function returns the Oracle Clusterware-generated resource ID. To retrieve the public display name of the resource, use the `clsagfw_get_resource_name()` API.

clsagfw_init()

The `clsagfw_init()` function initializes the agent framework.

Syntax

```
clsagfwret clsagfw_init(sb4 argc,
                        oratext **argv,
                        ub4 flags,
                        const oratext *logName,
                        void *reserved);
```

Input

`argc`: Number of command line arguments passed to the agent process

`argv`: Command line arguments passed to the agent process

`flags`:

logName: Name of the log file for this agent

reserved: Pass NULL

Notes

Oracle Clusterware automatically starts the agent configured for a resource when actions must be performed on the resource. On startup, the agent process calls this function first to initialize the agent framework run time. The command line arguments passed to the agent process by the Oracle Clusterware server must be passed to clsagfw_init(). After initialization is complete, the agent can register resource types that are managed by the agent.

clsagfw_is_cmd_timedout()

The clsagfw_is_cmd_timedout() function returns the cause of the ABORT entry point.

Syntax

```
bool clsagfw_is_cmd_timedout(const clsagfw_aectx *pAeCtx);
```

Input

pAeCtx: Agent framework context pointer that is passed to entry points

Notes

Agents can use this function in the ABORT entry point to identify the cause of the ABORT entry point invocation. The ABORT entry point can either be invoked if a command execution time exceeds its time out value or if the Oracle Clusterware administrator aborts the command.

clsagfw_log()

The clsagfw_log() function prints a diagnostic message to the agent log file.

Syntax

```
void clsagfw_log(const clsagfw_aectx *pAeCtx,
                const ub4 log_level,
                const oratext *fmtp, ...);
```

Input

pAeCtx: Agent framework context that is passed to user entry points

log_level: Log level of the message in the range of 1-5

fmtp: Message format string, according to Standard C library print specifications

. . .: Variable parameters corresponding to the format string

Notes

Agents call this function to log diagnostic messages that help in debugging and understanding agent execution. You can use the LOG_LEVEL parameter to control the volume of log messages. You can configure the current logging level of a resource using CRSCTL, as follows:

```
$ crsctl set log level res "myResource1=3"
```

The misusages are logged to the agent log file in the TODO directory. By default, the name of the log file is the same as the agent executable name. You can change this by passing the logName parameter to the clsagfw_init() API.

clsagfw_modify_attribute()

The clsagfw_modify_attribute() function modifies an attribute of the current resource.

Syntax

```
void clsagfw_modify_attribute(const clsagfw_aectx *pAeCtx,
                              const oratext *pAttrName,
                              const oratext *pAttrValue);
```

Input

pAeCtx: Agent framework context pointer that is passed to entry points

pAttrName: Name of attribute to be modified

pAttrValue: New value of attribute

Notes

Agents can modify the value of an attribute of the current command resource by calling this API. A resource modification request is sent to the Oracle Clusterware server for validation and the updated attribute value is reflected in the agent only after confirmation from the Oracle Clusterware server. This function can be only called from the START, STOP, and CLEAN entry points.

clsagfw_reset_attr_iterator()

The clsagfw_reset_attr_iterator() function resets the attribute iterator.

Syntax

```
clsagfwret clsagfw_reset_attr_iterator(const clsagfw_aectx *pAeCtx,
                                       clsagfw_ai_flags    flags);
```

Input

pAeCtx: Agent framework context pointer that is passed to entry points

flags: Which attributes to be accessed:

CLSAGFW_ALL_ATTRS: All attributes of the resource

CLSAGFW_MODIFIED_ATTRS: Only modified attributes

Notes

This function resets the iterator in the agent framework context so that agents can restart attribute navigation.

clsagfw_send_status2()

The clsagfw_send_status2() function reports progress, warning, and error messages to the Oracle Clusterware server and the user.

Syntax

```
void clsagfw_send_status2(const clsagfw_aectx* pAeCtx,
                          const oratext*       pResId,
                          clsagfw_status_type type,
                          const oratext* fmtp,...);
```

Input

`pAeCtx`: Agent framework context pointer that is passed to entry points

`pResId`: Resource for which the message is being reported

`type`: One of the following types of messages:

> `CLSAGFW_STATUS_PROGRESS`: Progress message
>
> `CLSAGFW_STATUS_WARNING`: Warning message
>
> `CLSAGFW_STATUS_ERROR`: Error message

`fmtp`: Message format string

. . .: Variables corresponding to positions in the message format string

Notes

During execution of a command, the agent can send informational messages to the Oracle Clusterware server and to the user performing operations on resources. These messages can inform you about execution progress and if there are any warnings or errors.

clsagfw_set_cookie()

The `clsagfw_set_cookie()` function stores a cookie and its value in the agent framework.

Syntax

```
clsagfwret clsagfwret clsagfw_set_cookie(const oratext *key,
                                         const void *value);
```

Input

`key`: Key of the cookie, null terminated string

`Value`: Pointer to the value of the cookie to be stored

Notes

The value of the saved cookie can be later retrieved by calling `clsagfw_get_cookie()`.

clsagfw_set_entrypoint()

The `clsagfw_set_entrypoint()` function sets the C/C++ entry point function to be executed for a particular action on the resource.

Syntax

```
void clsagfw_set_entrypoint(const oratext *type_name,
                            clsagfw_action_entry action_entry,
                            clsagfw_aecode action_entrycode)
```

Input

key: Key of the cookie, null terminated string

Value: Pointer to the value of the cookie to be stored

Notes

The value of the saved cookie can be later retrieved by calling clsagfw_get_cookie().

clsagfw_set_exitcb()

The clsagfw_set_exitcb() function registers a function to be called when the agent process exits.

Syntax

```
typedef void (*clsagfw_exit_callback) (sb4 exitCode);
void clsagfw_set_exitcb(clsagfw_exit_callback exit_func)
```

Input

exit_func: Function to be invoked when agent process exits

Notes

Agents can register a callback function to be invoked when the agent framework terminates the agent process. The callback function can perform proper clean up of any internal resources instantiated by the agent. If the agent process is exiting to recover from an irrecoverable error, then the Oracle Clusterware server does not restart the agent until the old incarnation has exited.

clsagfw_set_resource_state_label()

The clsagfw_set_resource_state_label() function sets a custom label on the current resource.

Syntax

```
void clsagfw_set_resource_state_label(const clsagfw_aectx *pAeCtx,
                                      const oratext *pLabel);
```

Input

pAeCtx: Agent framework context pointer that is passed to entry points

pLabel: Label to associate with the resource, null terminated string

Notes

Agents can use this function to set a descriptive label associated with the state of the resource. This label is reported in the STATE_DETAILS attribute of the resource.

clsagfw_startup()

The clsagfw_startup() function starts the agent framework.

Syntax

```
clsagfwret clsagfw_startup(void)
```

Notes

After the agent framework is initialized and the resource types and their entry points are registered, the agent calls this API to start the agent framework, which, in turn, starts the multithreaded infrastructure to monitor and perform actions on the resources.

On success, this function never returns to the calling thread. Entry points for the resource actions are invoked by the agent framework threads as and when required.

Agent Example

Example G–1 details an agent implementation to manage a resource that represents a file.

Example G–1 demoagent1.cpp

```
/* Copyright (c) 2004, 2012, Oracle and/or its affiliates.
All rights reserved. */

/*
 * NAME
 * Demoagent1.cpp: Demonstrates agent to monitor a file
 *
 * DESCRIPTION
 * This is a sample program that demonstrates an agent to monitor
 * a file. The agent has the following tasks:
 *   - On startup         : Create the file.
 *   - On shutdown        : Delete the file.
 *   - On check command   : Detect whether the file is present or not.
 *   - On clean command   : Delete the file.
 * This program can be used as a basis to build up more complicated
 * agents that manage real-world resources.
 *


#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <clsagfw.h>
```

Include the agent framework `clsagfw.h` header file.

```
#define TEST_TYPE1                     (oratext *) "HOTFILE_TYPE"
#define TEST_PATHNAME_ATTR             (oratext *) "PATH_NAME"


/*
 * NAME
 *    test_agent_exit
 * DESCRIPTION
 *    Function to clean up, called by framework whenever the agent
 *    process is about to terminate
 * PARAMS
 *    exitCode, which is an sb4 describing the reason for exit.
 * RETURNS
 *    void
 */
void
test_agent_exit(sb4 exitCode)
```

```
{
  clsagfw_log(NULL, 1, (oratext *)"Demo Agent is exiting..\n");
}
```

Exit callback function registered with the agent framework.

```
/*
 * NAME
 *    type1_start
 * DESCRIPTION
 *    Action entry point for the 'start' command. Creates the file for
 *    monitoring.
 * PARAMS
 *    Pointer to the agent framework context
 * RETURNS
 *    ub4 which could be CLSAGFW_AE_FAIL or CLSAGFW_AE_SUCCESS
 *    depending on whether the action failed or succeeded.
 */
ub4
type1_start(const clsagfw_aectx *ectx)
{
    ub4  ret = CLSAGFW_AE_FAIL;
    const oratext   *pResName  = NULL;
    const oratext   *pPathName = NULL;
    FILE *fp;

    clsagfw_log(ectx, 1, (oratext *)"Start action called..");

    /* Try to read the resource name */
    if (clsagfw_get_attrvalue(ectx, (oratext *)"NAME", &pResName) !=
            CLSAGFW_SUCCESS)
    {
      goto done;
    }

    /* Try to read the PATH_NAME attribute */
    if (clsagfw_get_attrvalue(ectx, TEST_PATHNAME_ATTR, &pPathName) !=
            CLSAGFW_SUCCESS)
    {
      goto done;
    }
```

Retrieve the resource name and the value of the `PATH_NAME` attribute using `clsagfw_get_attrvalue()`.

```
    clsagfw_log(ectx, 1,
                (oratext *)"Start action arguments: resName = %s,
                pathName = %s", pResName, pPathName);

    /* Try to create the file */
    fp = fopen((char *)pPathName, "w");

    if(!fp)
    {
      /* Could not create the file */
        clsagfw_log(ectx, 1, (oratext *) "START action for resource %s:
                    FAILED\n", pResName);
        ret = CLSAGFW_AE_FAIL;
    }
    else
    {
      /* Created the file successfully */
        clsagfw_log(ectx, 1, (oratext *) "START action for resource %s:
                    SUCCESS\n", pResName);
        ret = CLSAGFW_AE_SUCCESS;
      fclose(fp);
    }

done:
    return ret;
}
```

Diagnostic and error messages are logged to the agent log file by calling `clsagfw_log()`.

```
/*
 * NAME
 *    type1_stop
 * DESCRIPTION
 *    Action entry point for the 'stop' and 'clean' commands. Deletes
 *    the file being monitored.
 * PARAMS
 *    Pointer to agent framework context
 * RETURNS
 *    ub4 which could be CLSAGFW_AE_FAIL or CLSAGFW_AE_SUCCESS
 *    depending on whether the action failed or succeeded.
 */
ub4
type1_stop (const clsagfw_aectx *ectx)
{
    ub4             ret = CLSAGFW_AE_FAIL;
    const oratext   *pResName  = NULL;
    const oratext   *pPathName = NULL;

    clsagfw_log(ectx, 1, (oratext *)"Stop action called..");

    /* Try to read the resource name */
    if (clsagfw_get_attrvalue(ectx, (oratext *)"NAME", &pResName) !=
            CLSAGFW_SUCCESS)
    {
      clsagfw_log(ectx, 1, (oratext *)"STOP action: %s: Could not
                  read attribute\n", pResName);
      goto done;
    }

    /* Try to read the PATH_NAME attribute */
    if (clsagfw_get_attrvalue(ectx, TEST_PATHNAME_ATTR, &pPathName) !=
            CLSAGFW_SUCCESS)
    {
        pPathName = pResName;
    }

    clsagfw_log(ectx, 1,
                (oratext *)"Stop action arguments: resName = %s,
                pathName = %s", pResName, pPathName);

    /* Try to delete the file */
    if (remove((char *)pPathName))
    {
      /* Could not delete the file */
      clsagfw_log(ectx, 1, (oratext *)"STOP action for resource %s:
                  FAILED\n", pResName);
      ret = CLSAGFW_AE_FAIL;
    }
    else
    {
      /* Deleted the file successfully */
      clsagfw_log(ectx, 1, (oratext *)"STOP action for resource %s:
                  SUCCESS\n", pResName);
      ret = CLSAGFW_AE_SUCCESS;
    }

done:
    return ret;
}

/*
 * NAME
 *    type1_check
 * DESCRIPTION
 *    Action entry point for the 'check' command. Determines if the
 *    file exists.
 * PARAMS
 *    Pointer to agent framework context.
 * RETURNS
 *    ub4 which gives the status of the resource. Check the
 *    agent framework reference for details on this function's
 *     return codes.
 */
ub4
type1_check(const clsagfw_aectx *ectx)
{
    ub4             ret = CLSAGFW_UNKNOWN;
    const oratext   *pResName  = NULL;
    const oratext   *pPathName = NULL;
    FILE *fp;

    clsagfw_log(ectx, 1, (oratext *)"Check action called..");

    /* Try to read the resource name */
    if (clsagfw_get_attrvalue(ectx, (oratext *) "NAME", &pResName) !=
            CLSAGFW_SUCCESS)
    {
      goto done;
    }

    /* Try to read the PATH_NAME attribute */
    if (clsagfw_get_attrvalue(ectx, TEST_PATHNAME_ATTR, &pPathName) !=
            CLSAGFW_SUCCESS)
    {
      clsagfw_log(ectx, 1, (oratext *)"CHECK action: %s: Could not
                  read attribute\n", pResName);
      goto done;
    }

    clsagfw_log(ectx, 1,
                (oratext *)"Check action arguments: resName = %s,
                pathName = %s", pResName, pPathName);

    /* Check if the file is accessible */
    fp = fopen((char *)pPathName, "r");
    if (!fp)
    {
      /* Could not open file */
      clsagfw_log(ectx, 1, (oratext *)"CHECK action: %s status -
                  UNPLANNED_OFFLINE\n", pResName);
      ret = CLSAGFW_UNPLANNED_OFFLINE;
    }
    else
    {
      /* Opened file successfully */
      clsagfw_log(ectx, 1,(oratext *)"CHECK action: %s status -
                  ONLINE\n", pResName);
      fclose(fp);
      ret = CLSAGFW_ONLINE;
    }

done:
    return ret;
}

/*
 * Initialization of the agent framework and registration of types is
 *  done in main.
 */
int main(sb4 argc, oratext **argv)
{
    clsagfw_log(NULL, 1, (oratext *)" *** Agent Framework Demo Agent
                Started *** \n");

    /*
     * Initialize the agent framework
     */
    if (clsagfw_init(argc, argv, 0, NULL, 0)
        != CLSAGFW_SUCCESS)
    {
      clsagfw_log(NULL, 1, (oratext *)"Failed to initilize the agent
                  framework\n");
      clsagfw_exit(-1);
    }

    /*
     * Set the exit callback function
     */
    clsagfw_set_exitcb(test_agent_exit);

    /*
     * Add the type definition to the framework
     */
    if (clsagfw_add_type(TEST_TYPE1) != CLSAGFW_SUCCESS)
    {
      clsagfw_log(NULL, 1,(oratext *)"Failed in adding type %s to the
                  framework\n", TEST_TYPE1);
      clsagfw_exit(-1);
    }


    /*
     * Set all entry points for for HOTFILE_TYPE
     */
    clsagfw_set_entrypoint(TEST_TYPE1, type1_start,
                           CLSAGFW_ACTION_START);
    clsagfw_set_entrypoint(TEST_TYPE1, type1_stop,
                           CLSAGFW_ACTION_STOP);
    clsagfw_set_entrypoint(TEST_TYPE1, type1_check,
                           CLSAGFW_ACTION_CHECK);
    clsagfw_set_entrypoint(TEST_TYPE1, type1_stop,
                           CLSAGFW_ACTION_CLEAN);

    clsagfw_log(NULL, 1, (oratext *)"Added resource type [%s] to the
                agent framework\n", TEST_TYPE1);

    /*
     * All set to go, Start the framework. This function does not
     * return if the framework starts successfully.
     */
    clsagfw_startup();

    /*** NOT REACHED **/

    return 0;
}
```

H

Oracle Clusterware C Application Program Interfaces

This appendix describes the Oracle Clusterware C application program interfaces (APIs). This appendix contains the following topics:

- About the Programming Interface (C API) to Oracle Clusterware
- Interactive CLSCRS APIs
- Non-Interactive CLSCRS APIs
- What-If APIs
- Server Categorization APIs
- STAT3 API
- Miscellaneous APIs

> **See Also:** Chapter 9, "Making Applications Highly Available Using Oracle Clusterware" for detailed information about using Oracle Clusterware to make applications highly available

About the Programming Interface (C API) to Oracle Clusterware

This section contains information about using the programming interface (C API) to Oracle Clusterware (CLSCRS).

- Overview
- Operational Notes
- Deprecated CLSCRS APIs
- Changes to Existing CLSCRS APIs

Overview

CLSCRS is a set of C-based APIs for Oracle Clusterware. The CLSCRS APIs enable you to manage the operation of entities that are managed by Oracle Clusterware. These entities include resources, resource types, servers, and server pools. You use the APIs to register user applications with Oracle Clusterware so that the clusterware can manage them and maintain high availability. Once an application is registered, you can manage it and query the application's status. If you no longer need the application, then you can stop it and unregister it from Oracle Clusterware.

Oracle Clusterware services are provided by Cluster Ready Services that runs as part of Oracle Clusterware. The CLSCRS API uses a context that is explicitly named in all function calls. The API does not store anything at the process or thread level. You can use the callbacks for diagnostic logging.

Note: You can install the Oracle Clusterware high availability API from the Oracle Database client installation media.

Operational Notes

This section includes the following topics:

- Context Initialization and Persistence
- Threading Support
- CLSCRS API Data Structures
- Memory Management
- Error Handling and Tracing
- Callback Mechanism
- Filters
- Script Agent Usage
- Help Interface

Context Initialization and Persistence

To use the CLSCRS APIs, you must first initialize the `clscrs` context. The calls to create and terminate this context are:

- `clscrs_init_crs`: Initializes the `clscrs` context
- `clscrs_term_crs`: Terminates the `clscrs` context

The caller is responsible for terminating the context when it is no longer needed.

Threading Support

If initialized with the `CLSCRS_FLAG_USETHREADS` flag, then the CLSCRS API may spawn threads internally. Every API function executes in the context of the calling thread. The API context object may not be used concurrently by multiple threads. However, no thread-affinity on the part of the client is required. A process may create multiple API contexts and use those on different threads, subject to the one-thread-per-one-context-at-a-time rule.

CLSCRS API Data Structures

The following entities are passed into the API calls and contain return values from the API call:

- `clscrs_sp`: A *stringpair* (`sp`) contains a name and a value string. The value can be NULL. It is created and destroyed, and its contents can be examined and the value replaced. A stringpair can be a member of exactly one stringpair list (`splist`).
- `clscrs_splist`: A stringpair list (`splist`) is a list of zero or more stringpairs used in various contexts. An API can add stringpairs to or remove them from a stringpair list, or the API can iterate stringpairs.
- `clscrs_entity_type`: The enumeration type you can use to identify the type of Oracle Clusterware entity.

The enumeration types are defined, as follows:

```
/* enum to specify the entity type */
typedef enum
{
   clscrs_entity_res              = 1, /* resource */
   clscrs_entity_restype          = 2, /* resource type */
   clscrs_entity_serverpool       = 3, /* server pool */
   clscrs_entity_server           = 4, /* server */
   clscrs_entity_resinst          = 5, /* resource instances */
   clscrs_entity_config_policy    = 6, /* configuration policy */
   clscrs_entity_config_policyset = 7, /* configuration policy
set */
   clscrs_entity_server_category  = 8  /* server category */
} clscrs_entity_type;
```

- clscrs_crsentity: This data structure (crsentity) represents an Oracle Clusterware entity, which contains the name and additional data appropriate to the context in which the Oracle Clusterware entity is used. Sometimes an API contains Oracle Clusterware entity attribute data and other times it carries status and return messages about an operation. A single entity can be a member of exactly one clscrs_crsentitylist.
- clscrs_crsentitylist: An entity list (crsentitylist) is a data structure that contains zero or more instances of a clscrs_crsentity. An API can add entities to or remove them from an entity list, or the API can iterate entities.

Memory Management

The CLSCRS APIs work on elements and lists. The elements are added to lists. The memory for both elements and lists is allocated and released through explicit API calls. It is the caller's responsibility to release the memory that they allocate. However, when elements are added to lists, only the list must be destroyed: the destruction of the list destroys its elements implicitly. The elements must be destroyed when they are not added to any list. For recursive lists, destroying the parent list also destroys any lists contained within it. The clscrs_sp and clscrs_crsentity elements must be destroyed by the caller. If they are part of a clscrs_splist or clscrs_crsentitylist, destroying the list destroys the respective clscrs_sp and clscrs_crsentity entities.

For example, when a resource is created and added to a resource list, only the resource list must be destroyed, but not the individual resource. Destroying the resource list releases the memory for the individual resource, too.

Memory is allocated by the API through the following calls:

```
clscrs_sp_create()
clscrs_crsentity_create()
clscrs_serverpool_create()
clscrs_type_create()
clscrs_splist_create()
clscrs_crsentitylist_create()
clscrs_entity_id_create()
```

Each of the calls in the preceding list has a corresponding clscrs_*_destroy() call.

Error Handling and Tracing

Interactive and non-interactive CLSCRS APIs each use a different error-handling mechanism.

For non-interactive CLSCRS APIs, the error code is returned as the return value of the function call. For example:

```
clscrsret clscrs_sp_get_value(clscrs_sp *sp, oratext **value);
```

The error code is returned as a `clscrsret` value.

For interactive CLSCRS APIs, the output result is represented, as follows:

1. The return value of the function call provides a high-level output of the request. Did the request reach the server? Was it completely successful, or completely or only partially unsuccessful? A successful return value means the request was received, processed, and the outcome was successful for all entities requested.
2. For each entity on which the request operated, there is a programmatic completion code stored in the `op_status` list. If the value is not `success`, it indicates the high-level type of the problem specific to processing the request for the particular object.
3. Optionally, the API might indicate that it wants to receive localized, human-readable error, warning, or status messages by using the callback mechanism. Each invocation of the callback provides the message, message type (severity), and the ID of the object to which the callback invocation pertains.

For example:

```
CLSCRS_STAT clscrs_register_resource2(clscrs_crsentitylist *in_crsentitylist,
uword flags,
                                       clscrs_msgf2 msgf, void *msgarg,
                                       clscrs_crsentitylist *op_status);
```

1. The function returns an error code of value `CLSCRS_STAT`.
2. The CRSD sends error messages, warning messages, and progress messages back to the client through the `clscrs_msgf2` callback. The client must implement the callback to process these messages returned by the CRSD.
3. In previous Oracle Clusterware releases, the API also contained results of each operation on the Oracle Clusterware entities as part of the `op_status` list. You can access that information using the following API:

   ```
   clscrsret clscrs_entity_get_op_status(clscrs_entity *res, CLSCRS_STAT *status,
                                         oratext **msg);
   ```

   The `status` argument contains a status code about the CRSD operation on the Oracle Clusterware entity. Additionally, the `msg` argument contains a message from the CRSD about the result of the operation. Though the `op_status` list continues to contain the results of the CRSD operation for each Oracle Clusterware entity in the `msg` argument, usage of the `msg` argument to get the error codes and messages has now been deprecated and is not supported for any use of the API on a new entity. Only pre-existing use cases (for acting on resources, specifically) are supported. Use the callback function to process any messages returned by the CRSD.

Callback Mechanism

Interactive CLSCRS APIs provide a callback mechanism that the clients can use to process error messages, warning messages, and progress messages sent by the CRSD.

The signature of the callback mechanism is:

```
typedef  void (*clscrs_msgf2)(void *usrp, const oratext *id, const oratext *msg,
                 clscrs_msgtype msgtype);
```

In the preceding syntax:

- usrp: Is a user-supplied pointer that probably contains the context of the call
- id: Is the identifier of the entity to which the message corresponds
- msg: Is the output text
- msgtype: Is the type of the message; either error, warning, or progress

Example H–1 describes an example of the callback mechanism.

Example H–1 Callback Mechanism

```
void myCallback(void *arg, const oratext *pId, const oratext *pMsg,
                clscrs_msgtype msgType)
{
    if (pMsg != NULL)
   {
       cout << pMsg << endl;
    }
}
```

Example H–2 describes how to use the callback mechanism in an interactive API.

Example H–2 Using the Callback Mechanism In an Interactive API

```
clscrs_start_resource2(pResIdList, NULL,
                       env, myCallback, NULL,
                       0, pOpStatus);
```

You can also print debug trace messages for the API, itself by passing the CLSCRS_FLAG_TRACE flag when creating the context. The signature for context creation is:

```
CLSCRS_STAT clscrs_init_crs(clscrs_ctx **ctx, clscrs_msgf2 errf, void *errCtx,
                            ub4 flags);
```

For the trace messages to work, you must specify both the CLSCRS_FLAG_TRACE flag and a clscrs_msgf2 callback mechanism in the clscrs_init_crs API.

The clscrs_msgf2 callback mechanism has the following signature:

```
typedef  void (*clscrs_msgf)(void *usrp, const oratext *msg, sword msglen);
```

Filters

You can use filters to narrow down Oracle Clusterware entities upon which a CLSCRS API operates. Simple filters are attribute-value pairs with an operator. Operators must be surrounded by spaces, as shown in the examples. You can combine simple filters into expressions called expression filters using Boolean operators.

Supported filter operators are:

=
>
<
!=
co: Contains
st: Starts with
en: Ends with

Supported Boolean operators are AND and OR.

Examples of filters are:

- `TYPE = type1`
- `((TYPE = type1) AND (CHECK_INTERVAL > 50))`
- `(TYPE = type1) AND ((CHECK_INTERVAL > 30) OR (AUTO_START co never))`
- `NAME en network.res`
- `TYPE st ora.db`

> **See Also:** Use the `clscrs_comparator enum` and the `clscrs_operator enum` located in the `$ORA_CRS_HOME/crs/demo/clscrsx.h` file to get the correct type for the above comparators and operators, respectively, in the API calls

There are two types of filters and CLSCRS has a set of APIs to create these filters:

- **Comparison filter**: A simple filter that compares two values. For example:

  ```
  TYPE = ora.db.type
  ```

  Use the `clscrs_compfilter_create` API to create a comparison filter. For example, to create the `(TYPE = ora.db.type)` comparison filter:

  ```
  clscrs_compfilter_create(ctx, clscrs_TYPE,
                    clscrs_comparator_eq, (const oratext *)"ora.db.type",
                    &myCompFilter);
  ```

- **Expression filter**: A filter that is created from either a set of comparison filters or expression filters, or both. For example:

  ```
  ((TYPE = ora.db.type) AND (CHECK_INTERVAL > 50))
  ```

  Use the `clscrs_expfilter_create` API to create a comparison filter. For example, to create an expression filter:

  ```
  clscrs_exprfilter_create(myCompFilter1, clscrs_operator_or,
                          myCompFilter2, &myExprFilter);
  ```

> **See Also:** The `$ORA_CRS_HOME/crs/demo/clscrsx.h` file for usage information for the `clscrs_compfilter_create` and `clscrs_expfilter_create` APIs

> **Note:** Both the `clscrs_compfilter_create` and `clscrs_expfilter_create` APIs allocate memory that must be freed by calling `clscrs_filter_destroy()`.

You can use filters in the following interactive CLSCRS APIs in place of an entity list:

```
clscrs_start_resource2
clscrs_stat2
clscrs_stop_resource2
clscrs_check_resource2
clscrs_relocate_resource2
```

Example H–3 describes using filters in an interactive CLSCRS API.

Example H–3 Filters In an Interactive CLSCRS API

```
clscrs_start_resource2(myCompFilter, NULL,
                       env, msgf2, NULL,
                       0, pOpStatus);
```

Script Agent Usage

When you use CLSCRS APIs inside script agent entry points, keep the following in mind:

1. Some actions, such as start, stop, and clean, are executed under a lock on the resource instance. Thus, issuing a request to the server to act on the resource directly or by extension of a relation results in a dead-lock.
2. Issuing read-only (`clscrs_stat2`) is generally safe unless it is an initial check, where the script agent must not call back on the server, because that results in a dead-lock, as well. Use the `clsagfw` APIs to query the check entry point.

> **See Also:** Appendix B, "Oracle Clusterware Resource Reference" for examples of script agents

Help Interface

You can find the entire list of CLSCRS APIs, including usage information for each, in the `$ORA_CRS_HOME/crs/demo/clscrsx.h` file, along with a demo called `crsapp.c`.

Deprecated CLSCRS APIs

Table H–1 lists the deprecated CLSCRS APIs and the corresponding replacement APIs for Oracle Clusterware.

Table H–1 Deprecated CLSCRS APIs

Deprecated API	Replacement
`clscrs_check_resource`	`clscrs_check_resource2`
`clscrs_entity`	`clscrs_entity_type`
`clscrs_fail_resource`	No replacement
`clscrs_msgf`	`clscrs_msgf2`
`clscrs_register_resource`	`clscrs_register_resource2`
`clscrs_relocate_resource`	`clscrs_relocate_resource2`
`clscrs_res_attr_count`	`clscrs_crsentity_attr_count`
`clscrs_res_create`	`clscrs_crsentity_create`
`clscrs_res_create`	`clscrs_crsentitylist_create`
`clscrs_res_destroy`	`clscrs_crsentity_destroy`
`clscrs_res_get_attr`	`clscrs_crsentity_get_attr`
`clscrs_res_get_attr_list`	`clscrs_crsentity_get_attr_list`
`clscrs_res_get_name`	`clscrs_crsentity_get_name`
`clscrs_res_get_node_list`	`clscrs_crsentity_get_node_list`
`clscrs_res_get_op_status`	`clscrs_crsentity_get_op_status`
`clscrs_res_get_reslist`	`clscrs_crsentity_get_crsentitylist`
`clscrs_res_set_attr`	`clscrs_crsentity_set_attr`

Table H–1 (Cont.) Deprecated CLSCRS APIs

Deprecated API	Replacement
`clscrs_res_set_attr_list`	`clscrs_crsentity_set_attr_list`
`clscrs_res_set_reslist`	`clscrs_crsentity_set_crsentitylist`
`clscrs_reslist_append`	`clscrs_crsentitylist_append`
`clscrs_reslist_count`	`clscrs_crsentitylist_count`
`clscrs_reslist_delete_res`	`clscrs_crsentitylist_delete_crsentity`
`clscrs_reslist_destroy`	`clscrs_crsentitylist_destroy`
`clscrs_reslist_find`	`clscrs_crsentitylist_find`
`clscrs_reslist_first`	`clscrs_crsentitylist_first`
`clscrs_reslist_next`	`clscrs_crsentitylist_next`
`clscrs_start_resource`	`clscrs_start_resource2`
`clscrs_stat`	`clscrs_stat2`
`clscrs_stop_resource`	`clscrs_stop_resource2`
`clscrs_unregister_resource`	`clscrs_unregister_resource2`

Changes to Existing CLSCRS APIs

Oracle has added the following flags to the `clscrs_stat2` API:

- `CLSCRS_STATFLAG_SERVERBYCATEGORY`: Use this flag to query the servers that match a particular server category.
- `CLSCRS_STATFLAG_CATEGORYBYSERVER`: Use this flag to query the server categories that match a particular server.

Oracle has added the following flags to the `clscrs_start_resource2` and `clscrs_stop_resource2` APIs:

- `CLSCRS_FLAG_T_HA_PREPARE`: Use this flag with Transparent HA (`start2/stop2` APIs) to instruct the Oracle Clusterware daemon (CRSD) to prepare for an external management interface to act on a resource. When the call comes back, if successful, then the interface then is expected to start or stop the resource and call back with the other flag (`CLSCRS_FLAG_T_HA_FINISH`).
- `CLSCRS_FLAG_T_HA_FINISH`: Use this flag with Transparent HA with `CLSCRS_FLAG_T_HA_PREPARE`. You must use this flag in the second call to the CRSD (start2/stop2 APIs) when the start or stop action has finished. Note that clients must indicate resource *instances* and never resources with this flag, because the CRSD must know to which instances the invocation of the API applies.
- `CLSCRS_NO_STATE_RESTORATION`: This flag is available for use to start and stop resource APIs and to instruct the CRSD to skip resource state restoration of the resources whose state was affected. That procedure is usually attempted unless you use this flag.

Interactive CLSCRS APIs

The APIs listed in Table H–2 make calls to the Cluster Ready Services daemon (CRSD) to run commands. The CRSD must be up and running for these APIs to function.

Table H–2 Summary of Interactive CLSCRS APIs for Oracle Clusterware

C API	Description
`clscrs_check_resource2`	Notifies Oracle Clusterware to run the check entry points for the identified resources.
`clscrs_get_server_by_category`	Obtains a list of servers that match a particular server category.
`clscrs_is_crs_admin`	Checks whether the user is an Oracle Clusterware administrator.
`clscrs_register_resource2`	Registers the resources in the input resource list.
`clscrs_register_servercategory`	Registers server categories in the input server category list.
`clscrs_register_serverpool`	Registers a server pool for the input list of servers.
`clscrs_register_type`	Registers the resource types in the input resource list.
`clscrs_relocate_resource2`	Relocates the list of resource identifiers.
`clscrs_relocate_server`	Relocates a list of servers.
`clscrs_request_action`	Notifies Oracle Clusterware to run a specific set of actions.
`clscrs_restart_resource`	Instructs Oracle Clusterware to restart a named set of resources.
`clscrs_start_resource2`	Notifies Oracle Clusterware to start a named set of resources.
`clscrs_stat2`	Obtains information about specific resources.
`clscrs_stat3`	Obtains information about specific entities.
`clscrs_stop_resource2`	Notifies Oracle Clusterware to stop a named set of resources.
`clscrs_stop_resource_in_pools`	Instructs Oracle Clusterware to stop specific resources in server pools.
`clscrs_start_resource_in_pools`	Instructs Oracle Clusterware to start specific resources in server pools.
`clscrs_unregister_resource2`	Unregisters the resources in the input list of resource names.
`clscrs_unregister_servercategory`	Unregisters server categories in the input server category list.
`clscrs_unregister_serverpool`	Unregisters the given server pool.
`clscrs_unregister_type`	Unregisters the resource types in the input list.
`clscrs_whatif_add_server`	Simulates what happens if you add a server.
`clscrs_whatif_delete_server`	Simulates what happens if you delete a server.
`clscrs_whatif_fail_resource`	Simulates what happens if a resource fails.
`clscrs_whatif_register_resource`	Simulates what happens if you register a resource.
`clscrs_whatif_register_serverpool`	Simulates what happens if you register a server pool.
`clscrs_whatif_relocate_resource`	Simulates what happens if you relocate a resource.
`clscrs_whatif_relocate_server`	Simulates what happens if you relocate a server.
`clscrs_whatif_set_activepolicy`	Simulates what happens if you activate a policy.
`clscrs_whatif_start_resource`	Simulates what happens if you start a resource.
`clscrs_whatif_stop_resource`	Simulates what happens if you stop a resource.
`clscrs_whatif_unregister_serverpool`	Simulates what happens if you unregister a server pool.

Non-Interactive CLSCRS APIs

You can use non-interactive CLSCRS APIs for functions such as context initialization, preparing request payloads for interactive APIs, and post-processing output of the interactive APIs. The non-interactive CLSCRS APIs do not call the CRSD.

A callback error reporting mechanism is not available for the non-interactive CLSCRS APIs. All interactive CLSCRS APIs, except `clscrs_stat2`, `clscrs_stat3`, and all `clscrs_whatif_*` APIs, use this callback mechanism. Clients of these APIs also use the callback mechanism to receive error, warning, and progress messages from the CRSD.

You can also use filters to reduce the list of Oracle Clusterware entities. You can also use filters in the interactive APIs to reduce the list of Oracle Clusterware entities.

Thread Safety

The routines provided to manage API data structures cannot be used with the same API context in multiple threads concurrently; however, no thread-affinity on the part of the client is required. If a separate API context is used in each instance, then a process may invoke these routines on multiple threads.

> **See Also:** "Error Handling and Tracing" on page H-3, "Callback Mechanism" on page H-4, and "Filters" on page H-5 for more information

Table H–3 describes the non-interactive CLSCRS APIs.

Table H–3 Non-Interactive CLSCRS APIs

C API	Description
`clscrs_action_getentity`	Returns the entity for the action.
`clscrs_action_getparams`	Returns the list of parameters for the action.
`clscrs_action_gettype`	Returns the type for the action.
`clscrs_actionlist_count`	Counts the number of actions in the action list.
`clscrs_actionlist_create`	Creates an action list.
`clscrs_actionlist_destroy`	Destroys the action list.
`clscrs_actionlist_first`	Returns the first action in an action list.
`clscrs_actionlist_next`	Returns the next action in an action list.
`clscrs_actionlist_print`	Prints the action list.
`clscrs_actionlist_seqid`	Returns the sequence ID for the action lists
`clscrs_compfilter_create`	Constructs a simple filter that compares two values.
`clscrs_crsentity_attr_count`	Counts the number of attributes for an entity.
`clscrs_crsentity_create`	Creates a new entity (allocates memory).
`clscrs_crsentity_destroy`	Destroys an entity and frees up memory.
`clscrs_crsentity_get_attr`	Obtains the value of an entity, server pool, or server attribute.
`clscrs_crsentity_get_attr_list`	Obtains the attribute list for an entity, resource type, server pool, or server.
`clscrs_crsentity_get_crsentitylist`	Obtains the entity list for an entity.
`clscrs_crsentity_get_name`	Obtains the name of an entity.

Table H–3 (Cont.) Non-Interactive CLSCRS APIs

C API	Description
`clscrs_crsentity_get_node_list`	Obtains a list of nodes currently hosting the entity.
`clscrs_crsentity_get_op_status`	Obtains the status of an operation for an entity.
`clscrs_crsentity_get_registered`	Obtains the registration status of an entity.
`clscrs_crsentity_set_attr`	Sets an attribute for an entity and a server pool.
`clscrs_crsentity_set_attr_list`	Sets the attribute list for an entity, resource type, server pool, or server.
`clscrs_crsentity_set_crsentitylist`	Sets the resource list for an entity.
`clscrs_crsentitylist_append`	Adds an entity to an entity list.
`clscrs_crsentitylist_count`	Counts the number of entities in an entity list.
`clscrs_crsentitylist_create`	Creates a list of entities.
`clscrs_crsentitylist_delete_crsentity`	Deletes an entity matching a given name from an entity list.
`clscrs_crsentitylist_destroy`	Destroys an entity list and frees up memory.
`clscrs_crsentitylist_find`	Finds an entity in an entity list matching a given name.
`clscrs_crsentitylist_first`	Obtains the first entity on an entity list.
`clscrs_crslist_next`	Obtains the current next entity from an entity list.
`clscrs_entity_id_create`	Creates an entity identifier that identifies an Oracle Clusterware entity such as a resource, resource type, server group, and so on.
`clscrs_entity_id_destroy`	Frees the memory associated with an entity identifier created from `clscrs_entity_id_create()`.
`clscrs_exprfilter_create`	Constructs an expression filter from comparison or expression filters, or both.
`clscrs_filter_destroy`	Frees the memory for a filter.
`clscrs_get_entity_type`	Obtains the entity type corresponding to the entity identifier provided.
`clscrs_get_fixed_attrlist`	Obtains the list of attributes that correspond to an attribute group identifier.
`clscrs_get_resource_instance_details`	Obtains the resource instance details, such as resource name, cardinality, and degree, from the resource instance identifier that is used.
`clscrs_getnodename`	Obtains the node name.
`clscrs_init_crs`	Initializes a context for communications with Oracle Clusterware.
`clscrs_sp_get`	Obtains the name and value components of a stringpair.
`clscrs_sp_get_value`	Obtains the value component of a stringpair.
`clscrs_sp_set`	Changes the value part of a stringpair.
`clscrs_splist_append`	Adds a new stringpair (`sp`) to a stringpair list (`splist`).

Table H–3 (Cont.) Non-Interactive CLSCRS APIs

C API	Description
`clscrs_splist_count`	Counts the number of stringpairs (`sp`) in a stringpair list (`splist`).
`clscrs_splist_create`	Creates a new stringpair list.
`clscrs_splist_create_and_set`	Creates a new stringpair list (`splist`) and set the name and value for the first stringpair in the list.
`clscrs_splist_delete_sp`	Deletes a stringpair (`sp`) from a stringpair list (`splist`).
`clscrs_splist_destroy`	Frees the memory for a stringpair list (`splist`).
`clscrs_splist_find`	Finds the value for a stringpair (`sp`) in a stringpair list (`splist`).
`clscrs_splist_first`	Obtains the first stringpair (`sp`) from a stringpair list (`splist`).
`clscrs_splist_next`	Obtains the current next stringpair (`sp`) from a stringpair list (`splist`). Current next stringpair is effectively the next stringpair in the stringpair list. The list iterator is stored within the API and is not exposed.
`clscrs_splist_replace`	Replaces the value for a stringpair (`sp`) in a stringpair list (`splist`).
`clscrs_term_crs`	Releases a context for communications with Oracle Clusterware.
`clscrs_type_create`	Creates a new resource type.
`clscrs_type_get_attr`	Obtains the value/properties of a resource type attribute.
`clscrs_type_set_attr`	Adds an attribute to a resource type.

What-If APIs

You can use the What-If APIs to predict Oracle Clusterware's response to a hypothetical planned or unplanned event. Oracle Clusterware can react to events in the system and produce a response action plan. This action plan consists of a series of resource state transitions or server pool reconfigurations, or both. The What-If APIs provide a mechanism to expose this action plan externally and to enable clients to predict the consequences of an event before it actually happens.

What-If response plans are available for the following event categories:

Resources: Start, stop, relocate, add, and modify
Server pools: Add, remove, and modify
Servers: Add, remove, and relocate
Policy: Change active policy
Server category: Modify

Oracle Clusterware provides What-If output as a list of *actions*, where each action represents a specific step performed by Oracle Clusterware. Each action is encapsulated by a `clscrs_action` structure, and the `clscrs_actionlist` structure represents the entire sequence of actions. Oracle Clusterware also provides a set of functions (`clscrs_action_*` for action structures and `clscrs_actionlist_*` for action lists) to create, examine, iterate over, and destroy these structures. Their usage is identical to that of the corresponding entity list and entity functions.

The What-If APIs also provide clients with the ability to make a query on the projected state of the system. The clscrs_querylist structure using the stat3 format specifies the query, and the clscrs_crsentitylist structure provides the result. Refer to the stat3 section for details on their usage.

Each What-If response that Oracle Clusterware provides includes a sequence ID, which indicates the current state of Oracle Clusterware. The sequence ID is incremented for every new event that Oracle Clusterware manages. Oracle Clusterware guarantees that, as long the sequence ID has not changed, the action plan provided will be executed, as is, for the event in question. For example, the action plan that Oracle Clusterware provides for a whatif start resource FOO request will be identical to the actions Oracle Clusterware takes take when an actual start resource FOO request is submitted, provided the sequence ID has not changed.

Example H–4 describes how you can use What-If APIs.

Example H–4 Sample Usage of What-If API

```
boolean             tracectx = TRUE;
oratext            *resid;
clscrs_ctx         *ctx;
clscrs_env          env;
clscrs_splist      *resid_list;
clscrs_action      *cur_actn;
clscrs_actionlist  *alist;
clscrs_splist      *params;

// Init crs
clscrs_init_crs(&ctx, (clscrs_msgf)clsuslztrace, &tracectx, (ub4)0);

// Init parameters to the call
clscrs_entity_id_create(ctx, "MYRES", clscrs_entity_res, &resid);
clscrs_splist_create(ctx, &resid_list);
clscrs_splist_append(resid_list, resid, NULL);
clscrs_actionlist_create(ctx, &alist);

// Make call into the what-if API
clscrs_whatif_start_resource(resid_list, nodename, flags, NULL, NULL, alist);

// Process the resulting list of actions
for(clscrs_actionlist_first(alist,&cur_actn);cur_actn;clscrs_actionlist_
next(alist,&cur_actn))
 {
  params = clscrs_action_getparams(cur_actn);
  switch(clscrs_action_gettype(cur_actn))
  {
   case clscrs_actiontype_resstate:
      // Read params and do something
         break;
   case clscrs_actiontype_srvmove:
      // Read params and do something
         break;
   case clscrs_actiontype_newgroup:
      // Read params and do something
         break;
   case clscrs_actiontype_errorcase:
      // Read params and do something
         break;
   }
  }
  clscrs_actionlist_destroy(alist);
  clscrs_splist_destroy(resid_list);
  clscrs_term_crs(&ctx);
```

Parameters for the APIs listed in this section are separated into those for which you provide input and those that display information when the function completes successfully.

clscrs_whatif_register_resource

The `clscrs_whatif_register_resource` API determines the actions that Oracle Clusterware would take if you add or modify a specific resource.

Parameters

Input:

`name`: Name of the resource.
`attrs`: The attributes of the specified resource.
`flags`: `CLSCRS_FLAG_NONE`
`CLSCRS_FLAG_WHATIF_VERBOSE`
`CLSCRS_FLAG_FORCE`
`CLSCRS_FLAG_REG_UPDATE` (to modify the resource)

See Also: `CLRSCRS_FLAG` enum for more details

`qlist`: Specifies the client query on Oracle Clusterware entity status. NULL indicates no query specified.

Output:

`status`: List containing the returned entity information.
`alist`: Action list containing the actions that Oracle Clusterware will perform.

Returns

`CLSCRS_STAT_SUCCESS`: Indicates that Oracle Clusterware completed the request successfully or that no entities matched a filter.
`CLSCRS_STAT_AUTHORIZATION_FAILURE`: Displays when authorization fails.
`CLSCRS_STAT_CONNECTION`: Displays if there is a communication error.
`CLSCRS_STAT_WRONG_ACTIVE_VERSION`: Displays if you use the API before you have upgraded the cluster to Oracle Clusterware 12*c*.
`CLSCRS_STAT_INTERNAL_ERROR`: Displays if an unexpected, non-user error occurs.
`CLSCRS_STAT_INVALID_ARGS`: Displays if you provide incorrect arguments.

CLSCRS_STAT

```
clscrs_whatif_register_resource(const oratext *name, clscrs_splist *attrs,
                                  uword flags, clscrs_querylist *qlist,
                                  clscrs_crsentitylist *status,
                                  clscrs_actionlist *alist);
```

clscrs_whatif_fail_resource

The `clscrs_whatif_fail_resource` API determines the actions that Oracle Clusterware would take if specific resources failed.

Parameters

Input:

name: Resource or instance ID, or a filter.
server: Name of the server on which the resource failure occurs. NULL is allowed.
flags: CLSCRS_FLAG_NONE
CLSCRS_FLAG_WHATIF_VERBOSE

See Also: CLRSCRS_FLAG enum for more details

qlist: Specifies the client query on Oracle Clusterware entity status. NULL indicates no query specified.

Output:

status: List containing the returned entity information.
alist: Action list containing the actions that Oracle Clusterware will perform.

Returns

CLSCRS_STAT_SUCCESS: Indicates that Oracle Clusterware completed the request successfully or that no entities matched a filter.
CLSCRS_STAT_AUTHORIZATION_FAILURE: Displays when authorization fails.
CLSCRS_STAT_CONNECTION: Displays if there is a communication error.
CLSCRS_STAT_WRONG_ACTIVE_VERSION: Displays if you use the API before you have upgraded the cluster to Oracle Clusterware 12*c*.
CLSCRS_STAT_INTERNAL_ERROR: Displays if an unexpected, non-user error occurs.
CLSCRS_STAT_INVALID_ARGS: Displays if you provide incorrect arguments.

CLSCRS_STAT

```
clscrs_whatif_fail_resource(clscrs_splist *name, const oratext *server,
                              uword flags, clscrs_querylist *qlist,
                              clscrs_crsentitylist *status,
                              clscrs_actionlist *alist);
```

clscrs_whatif_start_resource

The clscrs_whatif_start_resource API determines the actions that Oracle Clusterware would take if you started specific resources.

Parameters

Input:

name: Resource or instance ID, or a filter.
node: Name of the node on which you want to start the resource. NULL is allowed.
flags: CLSCRS_FLAG_NONE
CLSCRS_FLAG_WHATIF_VERBOSE
CLSCRS_FLAG_FORCE

See Also: CLRSCRS_FLAG enum for more details

qlist: Specifies the client query on Oracle Clusterware entity status. NULL indicates no query specified.

Output:

status: List containing the returned entity information.
alist: Action list containing the actions that Oracle Clusterware will perform.

Returns

CLSCRS_STAT_SUCCESS: Indicates that Oracle Clusterware completed the request successfully or that no entities matched a filter.
CLSCRS_STAT_AUTHORIZATION_FAILURE: Displays when authorization fails.
CLSCRS_STAT_CONNECTION: Displays if there is a communication error.
CLSCRS_STAT_WRONG_ACTIVE_VERSION: Displays if you use the API before you have upgraded the cluster to Oracle Clusterware 12*c*.
CLSCRS_STAT_INTERNAL_ERROR: Displays if an unexpected, non-user error occurs.
CLSCRS_STAT_INVALID_ARGS: Displays if you provide incorrect arguments.

CLSCRS_STAT

```
clscrs_whatif_start_resource(clscrs_splist *name, const oratext *node,
                              uword flags, clscrs_querylist *qlist,
                              clscrs_crsentitylist *status,
                              clscrs_actionlist *alist);
```

clscrs_whatif_stop_resource

The clscrs_whatif_stop_resource API determines the actions that Oracle Clusterware would take if you stopped specific resources.

Parameters

Input:

name: Resource or instance ID, or a filter.
flags: CLSCRS_FLAG_NONE
CLSCRS_FLAG_WHATIF_VERBOSE
CLSCRS_FLAG_FORCE

See Also: CLRSCRS_FLAG enum for more details

qlist: Specifies the client query on Oracle Clusterware entity status. NULL indicates no query specified.

Output:

status: List containing the returned entity information.
alist: Action list containing the actions that Oracle Clusterware will perform.

Returns

CLSCRS_STAT_SUCCESS: Indicates that Oracle Clusterware completed the request successfully or that no entities matched a filter.
CLSCRS_STAT_AUTHORIZATION_FAILURE: Displays when authorization fails.
CLSCRS_STAT_CONNECTION: Displays if there is a communication error.
CLSCRS_STAT_WRONG_ACTIVE_VERSION: Displays if you use the API before you have upgraded the cluster to Oracle Clusterware 12*c*.
CLSCRS_STAT_INTERNAL_ERROR: Displays if an unexpected, non-user error occurs.
CLSCRS_STAT_INVALID_ARGS: Displays if you provide incorrect arguments.

CLSCRS_STAT

```
clscrs_whatif_stop_resource(clscrs_splist *name, uword flags,
                              clscrs_querylist *qlist,
                              clscrs_crsentitylist *status,
                              clscrs_actionlist *alist);
```

clscrs_whatif_relocate_resource

The `clscrs_whatif_relocate_resource` API determines the actions that Oracle Clusterware would take if you relocated specific resources.

Parameters

Input:

`name`: Resource or instance ID, or a filter.
`flags`: `CLSCRS_FLAG_NONE`
`CLSCRS_FLAG_WHATIF_VERBOSE`
`CLSCRS_FLAG_FORCE`

> **See Also:** `CLRSCRS_FLAG` enum for more details

`qlist`: Specifies the client query on Oracle Clusterware entity status. NULL indicates no query specified.

Output:

`status`: List containing the returned entity information.
`alist`: Action list containing the actions that Oracle Clusterware will perform.

Returns

`CLSCRS_STAT_SUCCESS`: Indicates that Oracle Clusterware completed the request successfully or that no entities matched a filter.
`CLSCRS_STAT_AUTHORIZATION_FAILURE`: Displays when authorization fails.
`CLSCRS_STAT_CONNECTION`: Displays if there is a communication error.
`CLSCRS_STAT_WRONG_ACTIVE_VERSION`: Displays if you use the API before you have upgraded the cluster to Oracle Clusterware 12*c*.
`CLSCRS_STAT_INTERNAL_ERROR`: Displays if an unexpected, non-user error occurs.
`CLSCRS_STAT_INVALID_ARGS`: Displays if you provide incorrect arguments.

CLSCRS_STAT

```
clscrs_whatif_relocate_resource(clscrs_splist *name, const oratext *destnode,
                                uword flags, clscrs_querylist *qlist,
                                clscrs_crsentitylist *status,
                                clscrs_actionlist *alist);
```

clscrs_whatif_register_serverpool

The `clscrs_whatif_register_serverpool` API determines the actions that Oracle Clusterware would take if you registered a specific server pool.

Parameters

Input:

`name`: Name of the server pool.
`attrs`: The attributes of the specified server pool.
`flags`: `CLSCRS_FLAG_NONE`
`CLSCRS_FLAG_WHATIF_VERBOSE`
`CLSCRS_FLAG_FORCE`
`CLSCRS_FLAG_REG_UPDATE` (to modify the server pool)

> **See Also:** `CLRSCRS_FLAG` enum for more details

qlist: Specifies the client query on Oracle Clusterware entity status. NULL indicates no query specified.

Output:

status: List containing the returned entity information.
alist: Action list containing the actions that Oracle Clusterware will perform.

Returns

CLSCRS_STAT_SUCCESS: Indicates that Oracle Clusterware completed the request successfully or that no entities matched a filter.
CLSCRS_STAT_AUTHORIZATION_FAILURE: Displays when authorization fails.
CLSCRS_STAT_CONNECTION: Displays if there is a communication error.
CLSCRS_STAT_WRONG_ACTIVE_VERSION: Displays if you use the API before you have upgraded the cluster to Oracle Clusterware 12*c*.
CLSCRS_STAT_INTERNAL_ERROR: Displays if an unexpected, non-user error occurs.
CLSCRS_STAT_INVALID_ARGS: Displays if you provide incorrect arguments.

CLSCRS_STAT

```
clscrs_whatif_register_serverpool(const oratext *pname, clscrs_splist *attrs,
                                   uword flags, clscrs_querylist *qlist,
                                   clscrs_crsentitylist *status,
                                   clscrs_actionlist *alist);
```

clscrs_whatif_unregister_serverpool

The clscrs_whatif_unregister_serverpool API determines the actions that Oracle Clusterware would take if you unregistered a specific server pool.

Parameters

Input:

name: Name of the server pool.
flags: CLSCRS_FLAG_NONE
       CLSCRS_FLAG_WHATIF_VERBOSE
       CLSCRS_FLAG_FORCE

See Also: CLRSCRS_FLAG enum for more details

qlist: Specifies the client query on Oracle Clusterware entity status. NULL indicates no query specified.

Output:

status: List containing the returned entity information.
alist: Action list containing the actions that Oracle Clusterware will perform.

Returns

CLSCRS_STAT_SUCCESS: Indicates that Oracle Clusterware completed the request successfully or that no entities matched a filter.
CLSCRS_STAT_AUTHORIZATION_FAILURE: Displays when authorization fails.
CLSCRS_STAT_CONNECTION: Displays if there is a communication error.
CLSCRS_STAT_WRONG_ACTIVE_VERSION: Displays if you use the API before you have upgraded the cluster to Oracle Clusterware 12*c*.
CLSCRS_STAT_INTERNAL_ERROR: Displays if an unexpected, non-user error occurs.
CLSCRS_STAT_INVALID_ARGS: Displays if you provide incorrect arguments.

CLSCRS_STAT

```
clscrs_whatif_unregister_serverpool(const oratext *poolname, uword flags,
                                    clscrs_querylist *qlist,
                                    clscrs_crsentitylist *status,
                                    clscrs_actionlist *alist);
```

clscrs_whatif_add_server

The clscrs_whatif_add_server API determines the actions that Oracle Clusterware would take if you added a server.

Parameters

Input:

name: Name of the server.
attrs: The attributes of the specified server.
flags: CLSCRS_FLAG_NONE
       CLSCRS_FLAG_WHATIF_VERBOSE
       CLSCRS_FLAG_FORCE

See Also: CLRSCRS_FLAG enum for more details

qlist: Specifies the client query on Oracle Clusterware entity status. NULL indicates no query specified.

Output:

status: List containing the returned entity information.
alist: Action list containing the actions that Oracle Clusterware will perform.

Returns

CLSCRS_STAT_SUCCESS: Indicates that Oracle Clusterware completed the request successfully or that no entities matched a filter.
CLSCRS_STAT_AUTHORIZATION_FAILURE: Displays when authorization fails.
CLSCRS_STAT_CONNECTION: Displays if there is a communication error.
CLSCRS_STAT_WRONG_ACTIVE_VERSION: Displays if you use the API before you have upgraded the cluster to Oracle Clusterware 12*c*.
CLSCRS_STAT_INTERNAL_ERROR: Displays if an unexpected, non-user error occurs.
CLSCRS_STAT_INVALID_ARGS: Displays if you provide incorrect arguments.

CLSCRS_STAT

```
clscrs_whatif_add_server(const oratext *name, clscrs_splist *attrs, uword flags,
                         clscrs_querylist *qlist, clscrs_crsentitylist *status,
                         clscrs_actionlist *alist);
```

clscrs_whatif_relocate_server

The clscrs_whatif_relocate_server API determines the actions that Oracle Clusterware would take if you relocated a server to a different server pool.

Parameters

Input:

name: Name of the server.
topool: The name of the server pool to which you want to relocate the server.

```
flags: CLSCRS_FLAG_NONE
       CLSCRS_FLAG_WHATIF_VERBOSE
       CLSCRS_FLAG_FORCE
```

See Also: `CLRSCRS_FLAG` enum for more details

`qlist`: Specifies the client query on Oracle Clusterware entity status. NULL indicates no query specified.

Output:

`status`: List containing the returned entity information.
`alist`: Action list containing the actions that Oracle Clusterware will perform.

Returns

`CLSCRS_STAT_SUCCESS`: Indicates that Oracle Clusterware completed the request successfully or that no entities matched a filter.
`CLSCRS_STAT_AUTHORIZATION_FAILURE`: Displays when authorization fails.
`CLSCRS_STAT_CONNECTION`: Displays if there is a communication error.
`CLSCRS_STAT_WRONG_ACTIVE_VERSION`: Displays if you use the API before you have upgraded the cluster to Oracle Clusterware 12*c*.
`CLSCRS_STAT_INTERNAL_ERROR`: Displays if an unexpected, non-user error occurs.
`CLSCRS_STAT_INVALID_ARGS`: Displays if you provide incorrect arguments.

CLSCRS_STAT

```
clscrs_whatif_relocate_server(const oratext *name, const oratext *topool,
                              uword flags, clscrs_querylist *qlist,
                              clscrs_crsentitylist *status,
                              clscrs_actionlist *alist);
```

clscrs_whatif_delete_server

The `clscrs_whatif_delete_server` API determines the actions that Oracle Clusterware would take if you deleted a server.

Parameters

Input:

`name`: Name of the server.

```
flags: CLSCRS_FLAG_NONE
       CLSCRS_FLAG_WHATIF_VERBOSE
       CLSCRS_FLAG_FORCE
```

See Also: `CLRSCRS_FLAG` enum for more details

`qlist`: Specifies the client query on Oracle Clusterware entity status. NULL indicates no query specified.

Output:

`status`: List containing the returned entity information.
`alist`: Action list containing the actions that Oracle Clusterware will perform.

Returns

`CLSCRS_STAT_SUCCESS`: Indicates that Oracle Clusterware completed the request successfully or that no entities matched a filter.

CLSCRS_STAT_AUTHORIZATION_FAILURE: Displays when authorization fails.
CLSCRS_STAT_CONNECTION: Displays if there is a communication error.
CLSCRS_STAT_WRONG_ACTIVE_VERSION: Displays if you use the API before you have upgraded the cluster to Oracle Clusterware 12*c*.
CLSCRS_STAT_INTERNAL_ERROR: Displays if an unexpected, non-user error occurs.
CLSCRS_STAT_INVALID_ARGS: Displays if you provide incorrect arguments.

CLSCRS_STAT

```
clscrs_whatif_delete_server(const oratext *name, uword flags,
                           clscrs_querylist *qlist,
                           clscrs_crsentitylist *status,
                           clscrs_actionlist *alist);
```

clscrs_whatif_set_activepolicy

The clscrs_whatif_set_activepolicy API determines the actions that Oracle Clusterware would take if you activated a specific policy.

Parameters

Input:

name: Name of the policy.
flags: CLSCRS_FLAG_NONE
       CLSCRS_FLAG_WHATIF_VERBOSE
       CLSCRS_FLAG_FORCE

> **See Also:** CLRSCRS_FLAG enum for more details

qlist: Specifies the client query on Oracle Clusterware entity status. NULL indicates no query specified.

Output:

status: List containing the returned entity information.
alist: Action list containing the actions that Oracle Clusterware will perform.

Returns

CLSCRS_STAT_SUCCESS: Indicates that Oracle Clusterware completed the request successfully or that no entities matched a filter.
CLSCRS_STAT_AUTHORIZATION_FAILURE: Displays when authorization fails.
CLSCRS_STAT_CONNECTION: Displays if there is a communication error.
CLSCRS_STAT_WRONG_ACTIVE_VERSION: Displays if you use the API before you have upgraded the cluster to Oracle Clusterware 12*c*.
CLSCRS_STAT_INTERNAL_ERROR: Displays if an unexpected, non-user error occurs.
CLSCRS_STAT_INVALID_ARGS: Displays if you provide incorrect arguments.

CLSCRS_STAT

```
clscrs_whatif_set_activepolicy(const oratext *name, uword flags,
                              clscrs_querylist *qlist,
                              clscrs_crsentitylist *status,
                              clscrs_actionlist *alist);
```

Server Categorization APIs

Oracle Clusterware includes an entity called clsrcs_entity_server_category.

Parameters for the APIs listed in this section are separated into those for which you provide input and those that display information when the function completes successfully.

clscrs_servercategory_create

The clscrs_servercategory_create API creates a server category.

Parameters

Input:

ctx: CLSCRS context.
sc_name: Name of the server category.

Output:

sc: The newly created server category.

Returns

clscrsretSUCC: Indicates that Oracle Clusterware completed the request successfully.
clscrsretNOMEM: Displays if no memory can be allocated.
clscrsretBADCTX: Displays if the context is NULL.
clscrsretBADARG: Displays if the server name is NULL.

clscrsret

```
clscrs_servercategory_create(clscrs_ctx *ctx, const oratext *sc_name,
                                clscrs_crsentity **sc);
```

clscrs_servercategory_destroy

The clscrs_servercategory_destroy API frees memory for a server category.

Parameters

Input:

sc_name: Name of the server category you want to destroy to free up memory

Returns

clscrsretSUCC: Indicates that Oracle Clusterware completed the request successfully

clscrsret

```
clscrs_servercategory_destroy(clscrs_crsentity **sc);
```

clscrs_register_servercategory

The clscrs_register_servercategory API registers the server categories that you specify in the input server category list. The attributes for the server category are contained in the input server category list. The op_status list contains the results of the register operation for each server category and contains no valid attributes. The caller must create and populate the in_entitylist and must create the op_status list. Both of these lists must be destroyed by the caller. The op_status list cannot be reused with another API call. It must be created and destroyed for each API call.

One or more attributes of an already registered server category can be modified by passing the CLSCRS_FLAG_REG_UPDATE flag. The flags apply to all server categories in the input entity list.

Parameters

Input:

in_entitylist: The list of server categories you want to register.
flags: CLSCRS_FLAG_REG_UPDATE
CLSCRS_FLAG_QUEUE
CLSCRS_FLAG_FORCE
CLSCRS_FLAG_NONE

See Also: CLRSCRS_FLAG enum for more details

msgf: User message callback, which can be NULL.
msgarg: User callback argument, which can be NULL.

Output:

op_status: The entity list that holds the status of the register operation for each server category.

Returns

CLSCRS_STAT_SUCCESS: Indicates that all input server categories are successfully registered.
CLSCRS_STAT_FAILURE: Displays if at least one server category cannot be registered.
CLSCRS_STAT_CONNECTION: Displays if there is a communication error.
CLSCRS_STAT_WRONG_ACTIVE_VERSION: Displays if you use the API before you have upgraded the cluster to Oracle Clusterware 12*c*.
CLSCRS_STAT_INVALID_ARGS: Displays if any of the server categories in the input entity list do not have attributes.

CLSCRS_STAT

```
clscrs_register_servercategory(clscrs_crsentitylist *in_entitylist, uword flags,
                               clscrs_msgf2 msgf, void *msgarg,
                               clscrs_crsentitylist *op_status);
```

clscrs_unregister_servercategory

The clscrs_unregister_servercategory API unregisters the server categories that you specify in the input list. The op_status list contains the results of the unregister operation for each server category.

The caller must create and populate the sclist and must create the op_status list. Both of these lists must be destroyed by the caller. The op_status list cannot be reused with another API call and must be created and destroyed for each API call.

Parameters

Input:

sclist: The list of server categories you want to unregister.
flags: Specify option flags.
msgf: User message callback, which can be NULL.
msgarg: User callback argument, which can be NULL.

Output:

op_status: The entity list that holds the status of the unregister operation for each server category.

Returns

CLSCRS_STAT_SUCCESS: Indicates that all input server categories are successfully registered.
CLSCRS_STAT_FAILURE: Displays if at least one server category cannot be unregistered.
CLSCRS_STAT_CONNECTION: Displays if there is a communication error.
CLSCRS_STAT_WRONG_ACTIVE_VERSION: Displays if you use the API before you have upgraded the cluster to Oracle Clusterware 12*c*.

CLSCRS_STAT

```
clscrs_unregister_servercategory(clscrs_splist *sclist, uword flags,
                                  clscrs_msgf2 msgf, void *msgarg,
                                  clscrs_crsentitylist *op_status);
```

clscrs_get_server_by_category

The clscrs_get_server_by_category API gets a list of servers that match a particular server category.

Parameters

Input:

in_list: The list of server categories or a filter.

Output:

out_entitylist: Lists the servers matching the server category.

Returns

CLSCRS_STAT_SUCCESS: Indicates that all input server categories are successfully registered.
CLSCRS_STAT_FAILURE: Displays if at least one server category cannot be unregistered.
CLSCRS_STAT_CONNECTION: Displays if there is a communication error.

CLSCRS_STAT

```
clscrs_get_server_by_category(clscrs_splist *in_list,
                               clscrs_crsentitylist *out_entitylist);
```

clscrs_register_server

The clscrs_register_server API modifies the server attributes. The attributes for the server are contained in the input server list. The op_status list contains the results of the modify operation for each server but contains no valid attributes.

The caller must create and populate the in_entitylist and must create the op_status list. Both of these lists must be destroyed by the caller. The op_status list cannot be reused with another API call and it must be created and destroyed for each API call.

Note: Since Oracle currently supports, only, the CLSCRS_FLAG_REG_UPDATE flag will always need to be passed. The flags apply to all servers in the input entity list.

Parameters

Input:

in_entitylist: The list of server categories you want to register.
flags: CLSCRS_FLAG_QUEUE
       CLSCRS_FLAG_FORCE
       CLSCRS_FLAG_REG_UPDATE
       CLSCRS_FLAG_NONE

See Also: CLRSCRS_FLAG enum for more details

msgf: User message callback, which can be NULL.
msgarg: User callback argument, which can be NULL.

Output:

op_status: The entity list that holds the status of the register operation for each server.

Returns

CLSCRS_STAT_SUCCESS: Indicates that all input server categories are successfully registered.
CLSCRS_STAT_FAILURE: Displays if at least one server category cannot be registered.
CLSCRS_STAT_CONNECTION: Displays if there is a communication error.
CLSCRS_STAT_WRONG_ACTIVE_VERSION: Displays if you use the API before you have upgraded the cluster to Oracle Clusterware 12*c*.
CLSCRS_STAT_INVALID_ARGS: Displays if any of the server categories in the input entity list do not have attributes.

CLSCRS_STAT

```
clscrs_register_server(clscrs_crsentitylist *in_entitylist, uword flags,
                       clscrs_msgf2 msgf, void *msgarg,
                       clscrs_crsentitylist *op_status);
```

STAT3 API

Oracle Clusterware 11*g* release 2 (11.2) manages several entities, such as resources, server pools, and so on. However, the interfaces in that release only allowed retrieving (reading) entities by type, which meant that a single retrieval could only return entities of a single type. Therefore, clients that needed to get different types of entities and needed to have a consistent view of the data structures to make further decisions needed to rely on a work around using a special event sequence ID and, if necessary, reissue query requests several times (in theory, in a system with ever changing state/configuration such a solution is time-unbounded).

Oracle Clusterware 12*c* provides a mechanism to perform a consistent read of entities of several kinds. The mechanism works on entities transparently, such that addition of new managed entities do not require any changes to the mechanism.

This is achieved by the clscrs_stat3 API.

clscrs_stat3

The clscrs_stat3 API obtains information about the Oracle Clusterware entities identified in qlist. Information is returned in out_entitylist. Queries are executed such that the subsequent output set is consistent.

You create clscrs_query objects and append them to the clscrs_querylist object before passing the object to this function. Any attributes required for each entity type must be present in the clscrs_query object. If the attribute list is empty, then Oracle Clusterware returns all attributes.

The out_entitylist must be created and passed as an empty list. Any errors for an entity are returned in the out_entitylist.

The output is returned as a nested entity list. Results for individual queries are returned as respective entities in the out_entitylist. The output for individual queries is returned as an entity list for that specific entity. The type of results in the entity can be verified by calling clscrs_crsentity_get_type, to get the Oracle Clusterware entity type of the entity that is part of the out_entitylist.

For example, If you have two queries, one for resources and the other for resource types, then out_entitylist will contain two entity objects, one for each of the aforementioned Oracle Clusterware entity types. The entity list for each of these entity objects will contain the results of the queries. To determine what type of entity object a particular object is, you must call the clscrs_crsentity_get_type function on that entity object. If the query returns zero matches, then the size of the entity list for that entity object will be zero.

Note: Oracle supports only one clscrs_query object per entity type. If more than one clscrs_query object is created for the same entity type, then Oracle does not guarantee the stat3 API behavior.

Parameters

Input:

qlist: The list of Oracle Clusterware entities you want to query.

Output:

out_entitylist: The entity list that holds the returned entity information.

Returns

CLSCRS_STAT_SUCCESS: Indicates that the API successfully queried Oracle Clusterware.
CLSCRS_STAT_FAILURE: Displays if there is an error querying Oracle Clusterware.
CLSCRS_STAT_CONNECTION: Displays if there is a communication error.

CLSCRS_STAT

```
clscrs_stat3(clscrs_querylist *qlist,
             uword flags,
             clscrs_crsentitylist *out_entitylist);
```

Miscellaneous APIs

Parameters for the APIs listed in this section are separated into those for which you provide input and those that display information when the function completes successfully.

clscrs_get_error_details

The `clscrs_get_error_details` API retrieves and returns the `clsk` exception stack if there are any failures while invoking other CLSCRS APIs.

> **Note:** The caller is responsible for allocating memory for `error_buf`.

Parameters

Input:

`error_buf`: The buffer that will be populated with the error stack.
`buf_size`: The size of the buffer for `error_buf`. If the size of the buffer given is smaller than what is required, then the API returns a value for the necessary buffer size.

Returns

`clscrsretSUCC`: Indicates that the error stack printed successfully.
`clscrsretEMPTY`: Displays if the error stack is empty.
`clscrsretBADARG`: Displays if either `error_buf` or `buf_size` is NULL.
`clscrsretBUFFSMALL`: Displays if the buffer size that you specify is smaller than what is required.

clscrsret

```
clscrs_get_error_details(oratext* error_buf, size_t* buf_size);
```

clscrs_request_action

The `clscrs_request_action` API instructs Oracle Clusterware to run an action on a named set of resources. In the first argument, you can specify either a list of resource IDs or a filter. Specifying a filter enables the query and action to take place in a single call. The filter searches all registered resources.

Parameters

Input:

`action_name`: The name of the action to be performed.
`ridlist`: The list of resources or resource instance IDs to stop, or a filter.
`env`: Specify environment arguments to the stop operation.
`msgf`: User message callback, which can be NULL.
`msgarg`: User callback argument, which can be NULL.
`flag`: Either `async` or `queue` options.

Output:

`op_status`: The resource list that holds the status of the action operation for each resource.

Returns

`CLSCRS_STAT_INVALID_RESNAME`: Displays if `ridlist` is empty.
`CLSCRS_STAT_AUTHORIZATION_FAILURE`: Displays when authorization fails.
`CLSCRS_STAT_SUCCESS`: Displays if the request completes successfully for all requested entities or if no entities match a filter.
`CLSCRS_STAT_FAILURE`: Displays if at least one resource or resource ID does not stop successfully.
`CLSCRS_STAT_CONNECTION`: Displays if there is a communication error.

CLSCRS_STAT

```
clscrs_request_action(oratext *action_name, clscrs_splist *ridlist,
                      clscrs_env env, clscrs_msgf2 msgf,
                      void *msgarg, uword flag, clscrs_reslist *op_status);
```

clscrs_stop_resource_in_pools

The clscrs_stop_resource_in_pools API instructs Oracle Clusterware to stop a named set of resources in server pools. Functionality is similar to clscrs_stop_resource2, except that this function takes an extra argument—spoollist—which is a list of server pool names or a filter based on which a list of server pools is generated. This function *does not* take node as an argument.

Parameters

Input:

ridlist: The list of resources or resource instance IDs to stop, or a filter.
spoollist: The list of server pool names where a resource that you want to stop is running, or a filter.
env: Specify environment arguments to the stop operation.
msgf: User message callback, which can be NULL.
msgarg: User callback argument, which can be NULL.
flag: Either async, force, or event options.

Output:

op_status: The resource list that holds the status of the stop operation for each resource.

Returns

CLSCRS_STAT_INVALID_RESNAME: Displays if ridlist is empty.
CLSCRS_STAT_AUTHORIZATION_FAILURE: Displays when authorization fails.
CLSCRS_STAT_SUCCESS: Displays if the request completes successfully for all requested entities or if no entities match a filter.
CLSCRS_STAT_FAILURE: Displays if at least one resource or resource ID does not stop successfully.
CLSCRS_STAT_CONNECTION: Displays if there is a communication error.

CLSCRS_STAT

```
clscrs_stop_resource_in_pools(clscrs_splist *ridlist,clscrs_splist *spoollist,
                              clscrs_env env,clscrs_msgf2 msgf2, void *msgarg,
                              uword flags, clscrs_reslist *op_status);
```

clscrs_start_resource_in_pools

The clscrs_start_resource_in_pools API instructs Oracle Clusterware to start a named set of resources in server pools. Functionality is similar to clscrs_start_resource2, except that this function takes an extra argument—spoollist—which is a list of server pool names or a filter based on which a list of server pools is generated. This function *does not* take node as an argument.

Parameters

Input:

ridlist: The list of resources or resource instance IDs to start, or a filter.

spoollist: The list of server pool names where a resource that you want to start is running, or a filter.
env: Specify environment arguments to the start operation.
msgf: User message callback, which can be NULL.
msgarg: User callback argument, which can be NULL.
flag: Either async, force, or event options.

Output:

op_status: The resource list that holds the status of the start operation for each resource.

Returns

CLSCRS_STAT_INVALID_RESNAME: Displays if ridlist is empty.
CLSCRS_STAT_AUTHORIZATION_FAILURE: Displays when authorization fails.
CLSCRS_STAT_SUCCESS: Displays if the request completes successfully for all requested entities or if no entities match a filter.
CLSCRS_STAT_FAILURE: Displays if at least one resource or resource ID does not start successfully.
CLSCRS_STAT_CONNECTION: Displays if there is a communication error.

CLSCRS_STAT

```
clscrs_start_resource_in_pools(clscrs_splist *ridlist,clscrs_splist *spoollist,
                               clscrs_env env,clscrs_msgf2 msgf2, void *msgarg,
                               uword flags, clscrs_reslist *op_status);
```

clscrs_restart_resource

The clscrs_restart_resource API instructs Oracle Clusterware to restart a named set of resources. If the flag is async, then the msgf callback function you specify is never called. The API returns an OK status after initiating the call to Oracle Clusterware, and asynchronously executes the restarts.

If the flag is not async, and msgf is not NULL, then the API drives msgf one line at a time with collected output from the restart programs. An optional event flag maybe passed to indicate that this is not a request to perform an action, but rather a notification that the action has already started. The flag should only be used for a narrow set of co-managed resources.

In the first argument, you can specify either a list of resource IDs or a filter. Specifying a filter enables the query and action to take place in a single call. The filter searches all registered resources.

Parameters

Input:

ridlist: The list of resources or resource instance IDs to restart, or a filter.
env: Specify environment arguments to the restart operation.
msgf: User message callback, which can be NULL.
msgarg: User callback argument, which can be NULL.
flag: Either async, force, or event options.

Output:

op_status: The resource list that holds the status of the restart operation for each resource.

Returns

CLSCRS_STAT_INVALID_RESNAME: Displays if ridlist is empty.
CLSCRS_STAT_AUTHORIZATION_FAILURE: Displays when authorization fails.
CLSCRS_STAT_SUCCESS: Displays if the request completes successfully for all requested entities or if no entities match a filter.
CLSCRS_STAT_FAILURE: Displays if at least one resource or resource ID does not start successfully.
CLSCRS_STAT_CONNECTION: Displays if there is a communication error.
CLSCRS_STAT_WRONG_ACTIVE_VERSION: Displays if you use the API before you have upgraded the cluster to Oracle Clusterware 12*c*.

CLSCRS_STAT

```
clscrs_restart_resource(clscrs_splist *ridlist,
                        clscrs_env env, clscrs_msgf2 msgf, void *msgarg,
                        uword flag, clscrs_reslist *op_status);
```

I

Oracle Cluster Registry Utility Reference

This appendix describes the syntax of the Oracle Cluster Registry (OCR) configuration utility, OCRCONFIG, and troubleshooting OCR, which includes descriptions and usage information for OCRCHECK and OCRDUMP.

This appendix contains the following topics:

About OCRCONFIG

Use the `ocrconfig` command to manage OCR. Using this utility you can import, export, add, delete, restore, overwrite, backup, repair, replace, move, upgrade, or downgrade OCR.

Usage Information

- The OCRCONFIG executable is located in the `Grid_home/bin` directory
- The `ocrconfig` command syntax is as follows:

  ```
  ocrconfig -option
  ```

Using Utility Help

To display the help output for the OCRCONFIG utility:

```
ocrconfig -help
```

Privileges and Security

To use the OCRCONFIG utility you must be logged into the operating system as a user with administrative privileges.

Log Files

The OCRCONFIG utility creates a log file in `*Grid_home*/log/*host_name*/client`.

To change the amount of logging, edit the path in the `*Grid_home*/srvm/admin/ocrlog.ini` file.

OCRCONFIG Command Reference

This section lists OCRCONFIG commands.

ocrconfig -add

Use the `ocrconfig -add` command to add an OCR location to a shared file system or Oracle Automatic Storage Management (Oracle ASM) disk group. OCR locations that you add must exist, have sufficient permissions, and, in the case of Oracle ASM disk groups, must be mounted before you can add them.

Syntax

```
ocrconfig -add location_name
```

Usage Notes

- You must run this command as `root`.
- The *`location_name`* variable can be a file name or the name of an Oracle ASM disk group. For example:
  - `/oradbocfs/crs/data.ocr`: You must create an empty (0 byte) OCR location
  - `d:\oracle\mirror.ocr`: You must create an empty (0 byte) OCR location
  - `+newdg`: Ensure that the disk group exists and is mounted

    If you specify an Oracle ASM disk group, then the name of the disk group must be preceded by a plus sign (+).

See Also:

- *Oracle Grid Infrastructure Installation Guide* for information about creating OCR locations and setting correct permissions
- *Oracle Automatic Storage Management Administrator's Guide* for more information about Oracle ASM disk group management

Example

To add an OCR location to the default location in Oracle ASM, `data`:

```
# ocrconfig -add +data
```

ocrconfig -backuploc

Use the `ocrconfig -backuploc` command to specify an OCR backup directory location.

Syntax

```
ocrconfig [-local] -backuploc file_name
```

Usage Notes

- You must run this command as `root`.
- Use the `-local` option to specify an OLR backup directory location.
- The *`file_name`* variable can be a full directory path name that is accessible by all nodes. For an OCR backup location, the *`file_name`* variable can also be an Oracle ASM disk group. For example:
  - *`Grid_home`*`/cdata/cluster3/`: Ensure that the location exists
  - `d:\cdata\cluster3`: Ensure that the location exists

  - `+bkupdg`: Ensure that the Oracle ASM disk group exists and is mounted on all nodes

    If you specify an Oracle ASM disk group, then ensure that the name of the disk group is preceded by a plus sign (+).

    Using an Oracle ASM disk group as a backup location is only supported for OCR.

    You must migrate OCR to Oracle ASM before changing the OCR backup location to an Oracle ASM disk group.

    You can change the OCR backup location to an Oracle ASM disk group only if there is at least one Oracle ASM OCR location in a separate disk group.

- The default location for generating OCR backups on Linux or UNIX systems is `*Grid_home*/cdata/*cluster_name*`, where *`cluster_name`* is the name of your cluster. The Windows default location for generating OCR backups uses the same path structure.
- The default location for generating OLR backups on Linux or UNIX systems is `*Grid_home*/cdata/*host_name*`, where *`host_name`* is the name of the node on which the OLR resides that you want to back up. The Windows default location for generating OLR backups uses the same path structure.

Example

To specify an OCR backup location in a directory:

```
# ocrconfig -backuploc $Grid_home/cdata/cluster3
```

To specify an OCR backup location in an Oracle ASM disk group:

```
# ocrconfig -backuploc +bkupdg
```

ocrconfig -copy

Use the `ocrconfig -copy` command to copy an OCR backup file from one location to another.

Syntax

```
ocrconfig -copy source_file_name destination_file_name
```

Usage Notes

- You must run this command as `root`.
- The *`source_file_name`* must be a valid OCR backup file. For example:
  - `/oradbocfs/crs/BACKUP00.ocr`
  - `d:\oracle\BACKUP01.ocr`
  - `+bkupdg:/aimev1/ocrbackup/day.ocr.260.739756845`

  If you specify an Oracle ASM disk group, then ensure that the disk group exists and is mounted.

  Examples of valid *`destination_file_name`* values are:

  - `/oradbocfs/crs/mybkp01.ocr`
  - `d:\oracle\mybkp01.ocr`

  - `+bkupdg:mybkp01`

  Oracle ASM disk group names must be preceded by a plus sign (+).

Examples

To copy an OCR backup file from a file system to an Oracle ASM disk group:

```
# ocrconfig -copy /oradbocfs/crs/BACKUP00.ocr +bkupdg:mybkp01.ocr
```

To copy an OCR backup file from an Oracle ASM disk group to a file system:

```
$ ocrconfig -copy +bkupdg:/aimev1/ocrbackup/day.ocr.260.739756845
/oradbocfs/crs/mybkp01.ocr
```

To copy an OCR backup file from one Oracle ASM disk group to another Oracle ASM disk group:

```
$ ocrconfig -copy +bkupdg01:/aimev1/ocrbackup/day.ocr.260.739756845
+bkupdg02:mybkp01.ocr
```

ocrconfig -delete

Use the `ocrconfig -delete` command to remove an OCR device or file.

Syntax

```
ocrconfig -delete file_name
```

Usage Notes

- You must run this command as `root`.
- The *`file_name`* variable can be a file name or the name of an Oracle ASM disk group. For example:
  - `/oradbocfs/crs/data.ocr`
  - `d:\oracle\mirror.ocr`
  - `+olddg`

  If you specify an Oracle ASM disk group, then the name of the disk group must be preceded by a plus sign (+).
- The only configured Oracle ASM OCR location, which is on a different Oracle ASM disk group than the Oracle ASM OCR backup location, cannot be removed.

Example

To remove an OCR location:

```
# ocrconfig -delete +olddg
```

ocrconfig -downgrade

Use the `ocrconfig -downgrade` command to downgrade OCR to an earlier specified version.

Syntax

```
ocrconfig -downgrade [-version version_string]
```

Usage Notes

You must run this command as `root`.

Example

To downgrade OCR to an earlier version:

```
# ocrconfig -downgrade -version 11.1.0.7
```

ocrconfig -export

Use the `ocrconfig -export` command to export the contents of OCR to a target file.

Syntax

```
ocrconfig [-local] -export file_name
```

Usage Notes

- You must run this command as `root`.
- Use the `-local` option to export the contents of OLR.
- The *`file_name`* variable can be a full path name. For example:
  - `/oradbocfs/crs/data.ocr`
  - `d:\oracle\`

Example

To export the contents of OCR to a file:

```
# ocrconfig -export d:\tmp\a
```

ocrconfig -import

Use the `ocrconfig -import` command to import the contents of a target file into which you exported the contents of OCR back into OCR.

Syntax

```
ocrconfig [-local] -import file_name
```

Usage Notes

- You must run this command as `root`.
- Use the `-local` option to import the contents of OLR from a file.
- The *`file_name`* variable must be a full path name that is accessible by all nodes. For example:
  - `/oradbocfs/crs/data.ocr`
  - `d:\oracle\`
- You must shut down Oracle Clusterware before running this command.

Example

To import the contents a file back into OCR:

```
# ocrconfig -import d:\tmp\a
```

ocrconfig -manualbackup

Use the `ocrconfig -manualbackup` command to back up OCR on demand in the location you specify with the `-backuploc` option.

Syntax

```
ocrconfig [-local] -manualbackup
```

Usage Notes

- You must run this command as `root`.
- Use the `-local` option to perform a manual backup of OLR.

Example

To back up OCR:

```
# ocrconfig -manualbackup
```

ocrconfig -overwrite

Use the `ocrconfig -overwrite` command to overwrite an OCR configuration in the OCR metadata with the current OCR configuration information that is found on the node from which you run this command.

Syntax

```
ocrconfig -overwrite
```

Usage Notes

You must run this command as `root`.

Example

To overwrite an OCR configuration:

```
# ocrconfig -overwrite
```

ocrconfig -repair

Use the `ocrconfig -repair` command to repair an OCR configuration on the node from which you run this command. Use this command to add, delete, or replace an OCR location on a node that may have been stopped while you made changes to the OCR configuration in the cluster. OCR locations that you add must exist, have sufficient permissions, and, in the case of Oracle ASM disk groups, must be mounted before you can add them.

Syntax

```
ocrconfig -repair -add file_name | -delete file_name | -replace
current_file_name -replacement new_file_name
```

Usage Notes

- You must run this command as `root`.
- Oracle High Availability Services must be started to successfully complete the repair.

- The Cluster Ready Services daemon must be stopped before running `ocrconfig -repair`.
- The *`file_name`* variable can be a valid OCR and either an absolute path name of an existing file or the name of an Oracle ASM disk group. For example:
  - `/oradbocfs/crs/data.ocr`
  - `d:\oracle\mirror.ocr`
  - `+newdg/mycluster/OCRFILE/registry.255.842574125`

    If you specify an Oracle ASM disk group, then the name of the disk group must be preceded by a plus sign (+).

    To obtain the name of an Oracle ASM disk group, run the `ocrcheck -config -details` command on a node where the OCR configuration *does not* need repair.

    See Also:

    - *Oracle Grid Infrastructure Installation Guide* for information about creating OCRs and setting correct permissions
    - *Oracle Automatic Storage Management Administrator's Guide* for more information about Oracle ASM disk group management
    - "Using the OCRCHECK Utility" on page I-10 for information about the `ocrcheck -config -details` command
- You can only use one option with `ocrconfig -repair` at a time.
- Running this command only modifies the local configuration and it and only affects the current node.

Example

To repair an OCR configuration:

```
# ocrconfig -repair -delete +olddg
```

ocrconfig -replace

Use the `ocrconfig -replace` command to replace an OCR device or file on the node from which you run this command. OCR locations that you add must exist, have sufficient permissions, and, in the case of Oracle ASM disk groups, must be mounted before you can add them.

Syntax

```
ocrconfig -replace current_location_name -replacement new_location_name
```

Usage Notes

- You must run this command as `root`.
- The *`new_location_name`* variable can be a file name or the name of an Oracle ASM disk group. For example:
  - `/oradbocfs/crs/data.ocr`: You must create an empty (0 byte) OCR location
  - `d:\oracle\mirror.ocr`: You must create an empty (0 byte) OCR location
  - `+newdg`: Ensure that the disk group exists and is mounted

If you specify an Oracle ASM disk group, then the name of the disk group must be preceded by a plus sign (+).

See Also:

- *Oracle Grid Infrastructure Installation Guide* for information about creating OCRs and setting correct permissions
- *Oracle Automatic Storage Management Administrator's Guide* for more information about Oracle ASM disk group management

- You must have at least two OCR devices to use this command. If you do not have at least two OCR devices, then you must run the `ocrconfig -add` command to add a new OCR device followed by the `ocrconfig -delete` command to delete the OCR device you want to replace.

Example

To replace an OCR device or file:

```
# ocrconfig -replace /oradbocfs/crs/data.ocr -replacement +newdg
```

ocrconfig -restore

Use the `ocrconfig -restore` command to restore OCR from an automatically created OCR backup file.

Syntax

```
ocrconfig [-local] -restore file_name
```

Usage Notes

- You must run this command as `root`.
- If the OCR backup file is located in an Oracle ASM disk group, then ensure that the disk group exists and is mounted.
- Before running this command, ensure that the original OCR or OLR files exist. If the original file does not exist, then you must create an empty file.

Note: You cannot use a file generated by the `ocrconfig -export` to restore OCR. The file must be a manual or automatic backup.

- Use the `-local` option to restore a backup of OLR.
- Example file names are:
  - `/oradbocfs/crs/BACKUP00.ocr`
  - `d:\oracle\BACKUP01.ocr`
  - `+bkupdg:/aimev1/ocrbackup/day.ocr.260.739756845`
- If OCR is located on an Oracle ASM disk group, then ensure that the disk group exists and is mounted.

Example

To restore OCR from a file:

```
# ocrconfig -restore /oradbocfs/crs/BACKUP00.ocr
```

ocrconfig -showbackup

Use the ocrconfig -showbackup command to display all available backups. By default, this command displays information for both automatic and manual backups unless you specify auto or manual.

Syntax

```
ocrconfig [-local] -showbackup [auto | manual]
```

Usage Notes

- Use the -local option to show manual OLR backup information. The -local flag functions only with the manual option.
- You can optionally specify auto or manual to display information about only automatic backups or only manual backups, respectively:
  - auto: Displays information about automatic backups that Oracle Clusterware created in the past 4 hours, 8 hours, 12 hours, and in the last day and week.
  - manual: Displays information for, at most, the last five manual backups that you invoke using the ocrconfig -manualbackup command.

Example

This command displays manual backup information for OLR, similar to the following:

```
$ ocrconfig -local -showbackup manual

cc40524 2012/10/12 15:06:57 /mjk_has5/oracle/log/srvm/backup_20121012_150657.olr
```

ocrconfig -showbackuploc

Use the ocrconfig -showbackuploc command to display the backup location of the OCR and OLR backup files.

Syntax

```
ocrconfig [-local] -showbackuploc
```

Usage Notes

- You must run this command as root
- Use the -local option to display the OLR backup location

Example

The ocrconfig -showbackuploc command returns output similar to the following:

```
# ocrconfig -showbackuploc

The Oracle Cluster Registry backup location is [+bkupdg]

# ocrconfig -local -showbackuploc

The Oracle Local Registry backup location is [/gridhome/oracle/log/srvm]
```

ocrconfig -upgrade

Only root scripts use the ocrconfig -upgrade command to upgrade OCR from a previous version.

Troubleshooting Oracle Cluster Registry and Diagnostic Output

This section describes various methods for troubleshooting problems with OCR, and obtaining diagnostic information from the utilities used to manage OCR. You can use these utilities to troubleshoot OLR.

This section contains the following topics:

- Troubleshooting Oracle Cluster Registry
- Using the OCRCHECK Utility
- Using the OCRDUMP Utility to View Oracle Cluster Registry Content

Troubleshooting Oracle Cluster Registry

Table I–1 describes common OCR problems with corresponding resolution suggestions.

Table I–1 Common Oracle Cluster Registry Problems and Solutions

Problem	Solution
Not currently using OCR mirroring and would like to enable it.	Run the `ocrconfig` command with the `-replace` option.
OCR failed and you must replace it. Error messages in Oracle Enterprise Manager or OCR log file.	Run the `ocrconfig` command with the `-replace` option.
OCR has a misconfiguration.	Run the `ocrconfig` command with the `-repair` option as described.
You are experiencing a severe performance effect from OCR processing or you want to remove OCR for other reasons.	Run the `ocrconfig` command with the `-replace` option as described.
OCR has failed and before you can fix it, the node must be rebooted with only one OCR.	Run the `ocrconfig` with the `-repair` option to remove the bad OCR location. Oracle Clusterware cannot start if it cannot find all OCRs defined.

Using the OCRCHECK Utility

The OCRCHECK utility displays the version of the OCR's block format, total space available and used space, OCRID, and the OCR locations that you have configured. OCRCHECK performs a block-by-block `checksum` operation for all of the blocks in all of the OCRs that you have configured. It also returns an individual status for each file and a result for the overall OCR integrity check.

Note: Oracle supports using the `ocrcheck` command when, at a minimum, the Oracle Cluster Ready Services stack is OFFLINE on all nodes in the cluster because the command will run even if the stack is ONLINE but it can falsely indicate that the OCR is corrupt if the check happens while an update to OCR is underway.

See Also: "The Oracle Clusterware Technology Stack" on page 1-7 for more information about the Oracle Cluster Ready Services stack

Syntax

```
ocrcheck [-local] [-config] [-details] [-help]
```

You can run the `ocrcheck -help` command to display usage information about this utility.

The `-details` option displays detailed OCR configuration information, including names for Oracle ASM disk groups.

Examples

Example I–1 OCRCHECK Utility Sample Output

```
# ocrcheck

Status of Oracle Cluster Registry is as follows :
        Version                  :          4
        Total space (kbytes)     :     409568
        Used space (kbytes)      :      17012
        Available space (kbytes) :     392556
        ID                       :  467263828
        Device/File Name         : +ocrdg1
                                   Device/File integrity check succeeded
        Device/File Name         : +ocrdg2
                                   Device/File integrity check succeeded
                                   Device/File not configured
                                   Device/File not configured
                                   Device/File not configured
        Cluster registry integrity check succeeded
        Logical corruption check succeeded
```

Note: The logical corruption check is only performed if you run the `ocrcheck` command as `root`.

Example I–2 shows using `ocrcheck -local -config` command to obtain OLR information.

Note: Oracle supports using the `ocrcheck -local` command when both the Oracle Cluster Ready Services and Oracle High Availability Services stacks are OFFLINE on the local node.

See Also: "The Oracle Clusterware Technology Stack" on page 1-7 for more information about the Oracle Cluster Ready Services and Oracle High Availability Services stacks

Example I–2 Using OCRCHECK to Obtain OLR Information

```
$ ocrcheck -local -config

Oracle Local Registry configuration is :
        Device/File Name         : /scratch/oracle/crshome/cdata/rwsbg08.olr
```

Example I–3 shows using the `ocrcheck -config -details` command to obtain the location of OCR in an Oracle ASM disk group.

Example I–3 Using OCRCHECK to Obtain OCR Information

```
$ ocrcheck -config -details
```

```
Oracle Cluster Registry configuration is :
        Device/File Name         : +ocrvd11/mjk_1/OCRFILE/registry.255.842574113
        Device/File Name         : +ocrdv2/mjk_1/OCRFILE/registry.255.842574125
```

Example I–4 shows using the `ocrcheck -details` command to obtain a greater level of detail for the OCR configuration.

Example I–4 Using OCRCHECK to Obtain More OCR Information

```
$ ocrcheck -details

Status of Oracle Cluster Registry is as follows :
        Version                  :          4
        Total space (kbytes)     :     409568
        Used space (kbytes)      :      17012
        Available space (kbytes) :     392556
        ID                       :  467263828
        Device/File Name         :
+OCRVD11/mycluster/OCRFILE/registry.255.842574113
                                   Device/File integrity check succeeded
        Device/File Name         :
+OCRVD2/mycluster/OCRFILE/registry.255.842574125
                                   Device/File integrity check succeeded
                                   Device/File not configured
                                   Device/File not configured
                                   Device/File not configured
        Cluster registry integrity check succeeded
        Logical corruption check succeeded
```

Example I–5 shows an example of OCR located on a shared file system.

Example I–5 OCR Located on a Shared File System

```
$ ocrcheck -config

Oracle Cluster Registry configuration is :
        Device/File Name         : /nas/data.ocr
        Device/File Name         : /nas2/mirror.ocr
        Device/File Name         : /nas3/mirror2.ocr
.
.
.
Status of Oracle Cluster Registry is as follows :
        Version                  :          4
        Total space (kbytes)     :     409568
        Used space (kbytes)      :      17012
        Available space (kbytes) :     392556
        ID                       :  157485888
        Device/File Name         : /dev/raw/raw15
                                   Device/File integrity check succeeded
        Device/File Name         : /dev/raw/raw17
                                   Device/File integrity check succeeded
        Device/File Name         : /dev/raw/raw16
                                   Device/File integrity check succeeded
                                   Device/File not configured
                                   Device/File not configured
        Cluster registry integrity check succeeded
        Logical corruption check succeeded
```

OCRCHECK creates a log file in the `*Grid_home*/diag/crs/*host_name*/crs/trace` directory. To change the log level, edit the `*Grid_home*/srvm/admin/ocrlog.ini` file.

Using the OCRDUMP Utility to View Oracle Cluster Registry Content

This section explains how to use the OCRDUMP utility to view OCR and Oracle Local Registry (OLR) content for troubleshooting. The OCRDUMP utility enables you to view OCR and OLR contents by writing the content to a file or `stdout` in a readable format.

You can use several options for OCRDUMP. For example, you can limit the output to a key and its descendents. You can also write the contents to an XML file that you can view using a browser. OCRDUMP writes the OCR keys as ASCII strings and values in a data type format. OCRDUMP retrieves header information based on a best effort basis.

OCRDUMP also creates a log file in `*Grid_home*/log/*host_name*/client`. To change the log level, edit the `*Grid_home*/srvm/admin/ocrlog.ini` file.

To change the logging component, edit the entry containing the `comploglvl=` entry. For example, to change the log level of the `OCRAPI` component to 3 and to change the log level of the `OCRRAW` component to 5, make the following entry in the `ocrlog.ini` file:

```
comploglvl="OCRAPI:3;OCRRAW:5"
```

Note: Make sure that you have file creation privileges in the *Grid_home* directory before using the OCRDUMP utility.

This section includes the following topics:

- OCRDUMP Utility Syntax and Options
- OCRDUMP Utility Examples
- Sample OCRDUMP Utility Output

OCRDUMP Utility Syntax and Options

This section describes the OCRDUMP utility command syntax and usage. Run the `ocrdump` command with the following syntax where *file_name* is the name of a target file to which you want Oracle Database to write the Oracle Cluster Registry output and where *key_name* is the name of a key from which you want Oracle Database to write Oracle Cluster Registry subtree content:

```
$ ocrdump [file_name | -stdout] [-local] [-backupfile backup_file_name
[-keyname key_name] [-xml] [-noheader]
]
```

Table I–2 describes the OCRDUMP utility options and option descriptions.

Table I–2 OCRDUMP Options and Option Descriptions

Options	Description
file_name	The name of a file to which you want OCRDUMP to write output. By default, OCRDUMP writes output to a predefined output file named `OCRDUMPFILE`. The *file_name* option redirects OCRDUMP output to a file that you specify.

Table I–2 (Cont.) OCRDUMP Options and Option Descriptions

Options	Description
`-stdout`	Use this option to redirect the OCRDUMP output to the text terminal that initiated the program. If you do not redirect the output, OCRDUMP writes output to a predefined output file named `OCRDUMPFILE`.
`-local`	Use this option to dump the contents of OLR.
`-backupfile`	Use this option to view the contents of an OCR backup file. Use the `-local` option with this option to view the contents of an OLR backup file.
`backup_file_name`	The name of the backup file with the content you want to view. You can query the backups using the `ocrconfig -showbackup` command. If the backup file is located in an Oracle ASM disk group, then ensure that the disk group exists and is mounted.
`-keyname` *`key_name`*	The name of an Oracle Cluster Registry key whose subtree is to be dumped.
`-xml`	Use this option to write the output in XML format.
`-noheader`	Does not print the time at which you ran the command and when the Oracle Cluster Registry configuration occurred.

OCRDUMP Utility Examples

The following `ocrdump` utility examples extract various types of OCR information and write it to various targets:

```
ocrdump
```

Writes OCR content to a file called `OCRDUMPFILE` in the current directory.

```
ocrdump MYFILE
```

Writes OCR content to a file called `MYFILE` in the current directory.

```
ocrdump -stdout -keyname SYSTEM
```

Displays OCR content from the subtree of the key `SYSTEM` in the terminal window.

```
ocrdump -stdout -xml
```

Displays OCR content in the terminal window in XML format.

```
ocrdump -stdout -backupfile Grid_home/cdata/cluster_name/file_name
```

Displays the content of the in the *`Grid_home`*`/cdata/`*`cluster_name`*/*`file_name`* directory. You must run this command as `root` to be able to view all of the keys. Be sure to name the file appropriately so that it can be recognized by anyone as an OCR backup file, such as `BACKUP00.ocr`.

Sample OCRDUMP Utility Output

The following OCRDUMP examples show the `KEYNAME`, `VALUE TYPE`, `VALUE`, permission set (`user`, `group`, `world`) and access rights for two sample runs of the `ocrdump` command. The following shows the output for the `SYSTEM.language` key that has a text value of `AMERICAN_AMERICA.WE8ASCII37`.

```
[SYSTEM.language]
ORATEXT : AMERICAN_AMERICA.WE8ASCII37
SECURITY : {USER_PERMISSION : PROCR_ALL_ACCESS, GROUP_PERMISSION : PROCR_READ,
OTHER_PERMISSION : PROCR_READ, USER_NAME : user, GROUP_NAME : group}
```

The following shows the output for the SYSTEM.version key that has integer value of 3:

```
[SYSTEM.version]
UB4 (10) : 3
SECURITY : {USER_PERMISSION : PROCR_ALL_ACCESS, GROUP_PERMISSION : PROCR_READ,
OTHER_PERMISSION : PROCR_READ, USER_NAME : user, GROUP_NAME : group}
```

J

Troubleshooting Oracle Clusterware

This appendix introduces monitoring the Oracle Clusterware environment and explains how you can enable dynamic debugging to troubleshoot Oracle Clusterware processing, and enable debugging and tracing for specific components and specific Oracle Clusterware resources to focus your troubleshooting efforts.

This appendix includes the following topics:

- Monitoring Oracle Clusterware
- Oracle Clusterware Diagnostic and Alert Log Data
- Diagnostics Collection Script
- Rolling Upgrade and Driver Installation Issues
- Oracle Trace File Analyzer Collector
- Testing Zone Delegation
- Oracle Clusterware Alerts

Monitoring Oracle Clusterware

You can use various tools to monitor Oracle Clusterware. While Oracle recommends that you use Oracle Enterprise Manager to monitor the everyday operations of Oracle Clusterware, Cluster Health Monitor (CHM) monitors the complete technology stack, including the operating system, for the purpose of ensuring smooth cluster operations. Both tools are enabled, by default, for any Oracle cluster, and Oracle strongly recommends that you use them.

This section includes the following topics:

- Oracle Enterprise Manager
- Cluster Health Monitor
- OCLUMON Command Reference

Oracle Enterprise Manager

You can use Oracle Enterprise Manager to monitor the Oracle Clusterware environment. When you log in to Oracle Enterprise Manager using a client browser, the Cluster Database Home page appears where you can monitor the status of both Oracle Clusterware environments. Monitoring can include such things as:

- Notification if there are any VIP relocations

- Status of the Oracle Clusterware on each node of the cluster using information obtained through the Cluster Verification Utility (`cluvfy`)
- Notification if node applications (nodeapps) start or stop
- Notification of issues in the Oracle Clusterware alert log for the Oracle Cluster Registry, voting file issues (if any), and node evictions

The Cluster Database Home page is similar to a single-instance Database Home page. However, on the Cluster Database Home page, Oracle Enterprise Manager displays the system state and availability. This includes a summary about alert messages and job activity, and links to all the database and Oracle Automatic Storage Management (Oracle ASM) instances. For example, you can track problems with services on the cluster including when a service is not running on all of the preferred instances or when a service response time threshold is not being met.

You can use the Oracle Enterprise Manager Interconnects page to monitor the Oracle Clusterware environment. The Interconnects page shows the public and private interfaces on the cluster, the overall throughput on the private interconnect, individual throughput on each of the network interfaces, error rates (if any) and the load contributed by database instances on the interconnect, including:

- Overall throughput across the private interconnect
- Notification if a database instance is using public interface due to misconfiguration
- Throughput and errors (if any) on the interconnect
- Throughput contributed by individual instances on the interconnect

All of this information also is available as collections that have a historic view. This is useful with cluster cache coherency, such as when diagnosing problems related to cluster wait events. You can access the Interconnects page by clicking the Interconnect tab on the Cluster Database home page.

Also, the Oracle Enterprise Manager Cluster Database Performance page provides a quick glimpse of the performance statistics for a database. Statistics are rolled up across all the instances in the cluster database in charts. Using the links next to the charts, you can get more specific information and perform any of the following tasks:

- Identify the causes of performance issues.
- Decide whether resources must be added or redistributed.
- Tune your SQL plan and schema for better optimization.
- Resolve performance issues

The charts on the Cluster Database Performance page include the following:

- **Chart for Cluster Host Load Average**: The Cluster Host Load Average chart in the Cluster Database Performance page shows potential problems that are outside the database. The chart shows maximum, average, and minimum load values for available nodes in the cluster for the previous hour.
- **Chart for Global Cache Block Access Latency**: Each cluster database instance has its own buffer cache in its System Global Area (SGA). Using Cache Fusion, Oracle RAC environments logically combine each instance's buffer cache to enable the database instances to process data as if the data resided on a logically combined, single cache.
- **Chart for Average Active Sessions**: The Average Active Sessions chart in the Cluster Database Performance page shows potential problems inside the database.

Categories, called wait classes, show how much of the database is using a resource, such as CPU or disk I/O. Comparing CPU time to wait time helps to determine how much of the response time is consumed with useful work rather than waiting for resources that are potentially held by other processes.

- **Chart for Database Throughput**: The Database Throughput charts summarize any resource contention that appears in the Average Active Sessions chart, and also show how much work the database is performing on behalf of the users or applications. The Per Second view shows the number of transactions compared to the number of logons, and the amount of physical reads compared to the redo size for each second. The Per Transaction view shows the amount of physical reads compared to the redo size for each transaction. Logons is the number of users that are logged on to the database.

In addition, the Top Activity drop down menu on the Cluster Database Performance page enables you to see the activity by wait events, services, and instances. Plus, you can see the details about SQL/sessions by going to a prior point in time by moving the slider on the chart.

See Also: *Oracle Database 2 Day + Real Application Clusters Guide*

Cluster Health Monitor

The Cluster Health Monitor (CHM) detects and analyzes operating system and cluster resource-related degradation and failures. CHM stores real-time operating system metrics in the Oracle Grid Infrastructure Management Repository that you can use for later triage with the help of My Oracle Support should you have cluster issues.

This section includes the following CHM topics:

- CHM Services
- Collecting CHM Data

CHM Services

CHM consists of the following services:

- System Monitor Service
- Cluster Logger Service
- Oracle Grid Infrastructure Management Repository

System Monitor Service

There is one system monitor service on every node. The system monitor service (`osysmond`) is a real-time, monitoring and operating system metric collection service that sends the data to the cluster logger service. The cluster logger service receives the information from all the nodes and persists in an Oracle Grid Infrastructure Management Repository database.

Cluster Logger Service

There is one cluster logger service (OLOGGERD) for every 32 nodes in a cluster. Another OLOGGERD is spawned for every additional 32 nodes (which can be a sum of Hub and Leaf Nodes). If the cluster logger service fails (because the service is not able come up after a fixed number of retries or the node where it was running is down), then Oracle Clusterware starts OLOGGERD on a different node. The cluster logger service manages the operating system metric database in the Oracle Grid Infrastructure Management Repository.

Oracle Grid Infrastructure Management Repository

The Oracle Grid Infrastructure Management Repository:

- Is an Oracle database that stores real-time operating system metrics collected by CHM. You configure the Oracle Grid Infrastructure Management Repository during an installation of or upgrade to Oracle Clusterware 12*c* on a cluster.

  > **Note:** If you are upgrading Oracle Clusterware to Oracle Clusterware 12*c* and Oracle Cluster Registry (OCR) and the voting file are stored on raw or block devices, then you must move them to Oracle ASM or a shared file system before you upgrade your software.

- Runs on one node in the cluster (this must be a Hub Node in an Oracle Flex Cluster configuration), and must support failover to another node in case of node or storage failure.

  You can locate the Oracle Grid Infrastructure Management Repository on the same node as the OLOGGERD to improve performance and decrease private network traffic.

- Communicates with any cluster clients (such as OLOGGERD and OCLUMON) through the private network. Oracle Grid Infrastructure Management Repository communicates with external clients over the public network, only.
- Data files are located in the same disk group as the OCR and voting file.

  If OCR is stored in an Oracle ASM disk group called `+MYDG`, then configuration scripts will use the same disk group to store the Oracle Grid Infrastructure Management Repository.

  Oracle increased the Oracle Clusterware shared storage requirement to accommodate the Oracle Grid Infrastructure Management Repository, which can be a network file system (NFS), cluster file system, or an Oracle ASM disk group.

- Size and retention is managed with OCLUMON.

Collecting CHM Data

You can collect CHM data from any node in the cluster by running the `*Grid_home*/bin/diagcollection.pl` script on the node.

> **Notes:**
>
> - Oracle recommends that, when you run the `diagcollection.pl` script to collect CHM data, you run the script on all nodes in the cluster to ensure gathering all of the information needed for analysis.
> - You must run this script as a privileged user.

> **See Also:** "Diagnostics Collection Script" on page J-17 for more information about the `diagcollection.pl` script

To run the data collection script on only the node where the cluster logger service is running:

1. Run the following command to identify the node running the cluster logger service:

```
$ Grid_home/bin/oclumon manage -get master
```

2. Run the following command from a writable directory outside the Grid home as a privileged user on the cluster logger service node to collect all the available data in the Oracle Grid Infrastructure Management Repository:

   ```
   # Grid_home/bin/diagcollection.pl --collect
   ```

   On Windows, run the following commands:

   ```
   C:\Grid_home\perl\bin\perl.exe
   C:\Grid_home\bin\diagcollection.pl --collect
   ```

   The `diagcollection.pl` script creates a file called `chmosData_`*`host_name_time_stamp`*`.tar.gz`, similar to the following:

   ```
   chmosData_stact29_20121006_2321.tar.gz
   ```

To limit the amount of data you want collected, enter the following command on a single line:

```
# Grid_home/bin/diagcollection.pl --collect --chmos
   --incidenttime time --incidentduration duration
```

In the preceding command, the format for the `--incidenttime` argument is `MM/DD/YYYY24HH:MM:SS` and the format for the `--incidentduration` argument is `HH:MM`. For example:

```
# Grid_home/bin/diagcollection.pl --collect --crshome Grid_home
   --chmos --incidenttime 07/21/2013 01:00:00 --incidentduration 00:30
```

OCLUMON Command Reference

The OCLUMON command-line tool is included with CHM and you can use it to query the CHM repository to display node-specific metrics for a specified time period. You can also use OCLUMON to perform miscellaneous administrative tasks, such as changing the debug levels, querying the version of CHM, and changing the metrics database size.

This section details the following OCLUMON commands:

- `oclumon debug`
- `oclumon dumpnodeview`
- `oclumon manage`
- `oclumon version`

oclumon debug

Use the `oclumon debug` command to set the log level for the CHM services.

Syntax

```
oclumon debug [log daemon module:log_level] [version]
```

Parameters

Table J–1 oclumon debug Command Parameters

Parameter	Description

`log` *`daemon module`*`:`*`log_level`*	Use this option change the log level of daemons and daemon modules. Supported daemons are:
	`osysmond`
	`ologgerd`
	`client`
	`all`
	Supported daemon modules are:
	`osysmond`: `CRFMOND`, `CRFM`, and `allcomp`
	`ologgerd`: `CRFLOGD`, `CRFLDREP`, `CRFM`, and `allcomp`
	`client`: `OCLUMON`, `CRFM`, and `allcomp`
	`all`: `allcomp`
	Supported *`log_level`* values are `0`, `1`, `2`, and `3`.

`version`	Use this option to display the versions of the daemons.

Example

The following example sets the log level of the system monitor service (`osysmond`):

```
$ oclumon debug log osysmond CRFMOND:3
```

oclumon dumpnodeview

Use the `oclumon dumpnodeview` command to view log information from the system monitor service in the form of a node view.

A node view is a collection of all metrics collected by CHM for a node at a point in time. CHM attempts to collect metrics every five seconds on every node. Some metrics are static while other metrics are dynamic.

A node view consists of eight views when you display verbose output:

- **SYSTEM**: Lists system metrics such as CPU COUNT, CPU USAGE, and MEM USAGE
- **TOP CONSUMERS**: Lists the top consuming processes in the following format:

  `metric_name``: '`*`process_name`*`(`*`process_identifier`*`) `*`utilization`*`'`

- **PROCESSES**: Lists process metrics such as PID, name, number of threads, memory usage, and number of file descriptors
- **DEVICES**: Lists device metrics such as disk read and write rates, queue length, and wait time per I/O
- **NICS**: Lists network interface card metrics such as network receive and send rates, effective bandwidth, and error rates
- **FILESYSTEMS**: Lists file system metrics, such as total, used, and available space
- **PROTOCOL ERRORS**: Lists any protocol errors
- **CPUS**: Lists statistics for each CPU

You can generate a summary report that only contains the SYSTEM and TOP CONSUMERS views.

"Metric Descriptions" on page J-8 lists descriptions for all the metrics associated with each of the views in the preceding list.

> **Note:** Metrics displayed in the TOP CONSUMERS view are described in Table J–4, " PROCESSES View Metric Descriptions".

Example J–1 shows an example of a node view.

Syntax

```
oclumon dumpnodeview [[-allnodes] | [-n node1 node2 noden] [-last "duration"] |
[-s "time_stamp" -e "time_stamp"] [-i interval] [-v]] [-h]
```

Parameters

Table J–2 oclumon dumpnodeview Command Parameters

Parameter	Description
`-allnodes`	Use this option to dump the node views of all the nodes in the cluster.
`-n` *node1 node2*	Specify one node (or several nodes in a space-delimited list) for which you want to dump the node view.
`-last "`*duration*`"`	Use this option to specify a time, given in HH24:MM:SS format surrounded by double quotation marks (`""`), to retrieve the last metrics. For example: `"23:05:00"`
`-s "`*time_stamp*`" -e "`*time_stamp*`"`	Use the `-s` option to specify a time stamp from which to start a range of queries and use the `-e` option to specify a time stamp to end the range of queries. Specify time in YYYY-MM-DD HH24:MM:SS format surrounded by double quotation marks (`""`). `"2011-05-10 23:05:00"` **Note:** You must specify these two options together to obtain a range.
`-i` *interval*	Specify a collection interval, in five-second increments.
`-v`	Displays verbose node view output.
`-h`	Displays online help for the `oclumon dumpnodeview` command.

Usage Notes

- In certain circumstances, data can be delayed for some time before it is replayed by this command. For example, the `crsctl stop cluster -all` command can cause data delay. After running `crsctl start cluster -all`, it may take several minutes before `oclumon dumpnodeview` shows any data collected during the interval.
- The default is to continuously dump node views. To stop continuous display, use Ctrl+C on Linux and Windows.
- Both the local system monitor service (`osysmond`) and the cluster logger service (`ologgerd`) must be running to obtain node view dumps.

Examples

The following example dumps node views from `node1`, `node2`, and `node3` collected over the last twelve hours:

```
$ oclumon dumpnodeview -n node1 node2 node3 -last "12:00:00"
```

The following example displays node views from all nodes collected over the last fifteen minutes at a 30 second interval:

```
$ oclumon dumpnodeview -allnodes -last "00:15:00" -i 30
```

Metric Descriptions

This section includes descriptions of the metrics in each of the seven views that comprise a node view listed in the following tables.

Table J–3 SYSTEM View Metric Descriptions

Metric	Description
`#pcpus`	The number of physical CPUs
`#vcpus`	Number of logical compute units
`cpuht`	CPU hyperthreading enabled (Y) or disabled (N)
`chipname`	The name of the CPU vendor
`cpu`	Average CPU utilization per processing unit within the current sample interval (%).
`cpuq`	Number of processes waiting in the run queue within the current sample interval
`physmemfree`	Amount of free RAM (KB)
`physmemtotal`	Amount of total usable RAM (KB)
`mcache`	Amount of physical RAM used for file buffers plus the amount of physical RAM used as cache memory (KB) On Windows systems, this is the number of bytes currently being used by the file system cache **Note:** This metric is not available on Solaris.
`swapfree`	Amount of swap memory free (KB)
`swaptotal`	Total amount of physical swap memory (KB)
`hugepagetotal`	Total size of huge in KB **Note:** This metric is not available on Solaris or Windows systems.
`hugepagefree`	Free size of huge page in KB **Note:** This metric is not available on Solaris or Windows systems.
`hugepagesize`	Smallest unit size of huge page **Note:** This metric is not available on Solaris or Windows systems.
`ior`	Average total disk read rate within the current sample interval (KB per second)
`iow`	Average total disk write rate within the current sample interval (KB per second)
`ios`	I/O operation average time to serve I/O request
`swpin`	Average swap in rate within the current sample interval (KB per second) **Note:** This metric is not available on Windows systems.

Table J–3 (Cont.) SYSTEM View Metric Descriptions

Metric	Description
swpout	Average swap out rate within the current sample interval (KB per second) **Note:** This metric is not available on Windows systems.
pgin	Average page in rate within the current sample interval (pages per second)
pgout	Average page out rate within the current sample interval (pages per second)
netr	Average total network receive rate within the current sample interval (KB per second)
netw	Average total network send rate within the current sample interval (KB per second)
procs	Number of processes
procsoncpu	The current number of processes running on the CPU
rtprocs	Number of real-time processes
rtprocsoncpu	The current number of real-time processes running on the CPU
#fds	Number of open file descriptors *or* Number of open handles on Windows
#sysfdlimit	System limit on number of file descriptors **Note:** This metric is not available on either Solaris or Windows systems.
#disks	Number of disks
#nics	Number of network interface cards
nicErrors	Average total network error rate within the current sample interval (errors per second)

Table J–4 PROCESSES View Metric Descriptions

Metric	Description
name	The name of the process executable
pid	The process identifier assigned by the operating system
#procfdlimit	Limit on number of file descriptors for this process **Note:** This metric is not available on Windows, AIX, and HP-UX systems.
cpuusage	Process CPU utilization (%) **Note:** The utilization value can be up to 100 times the number of processing units.
privmem	Process private memory usage (KB)
shm	Process shared memory usage (KB) **Note:** This metric is not available on Windows, Solaris, and AIX systems.
workingset	Working set of a program (KB) **Note:** This metric is only available on Windows.

Table J–4 (Cont.) PROCESSES View Metric Descriptions

Metric	Description
`#fd`	Number of file descriptors open by this process *or* Number of open handles by this process on Windows
`#threads`	Number of threads created by this process
`priority`	The process priority
`nice`	The nice value of the process **Note:** This metric is not applicable to Windows systems.
`state`	The state of the process **Note:** This metric is not applicable to Windows systems.

Table J–5 DEVICES View Metric Descriptions

Metric	Description
`ior`	Average disk read rate within the current sample interval (KB per second)
`iow`	Average disk write rate within the current sample interval (KB per second)
`ios`	Average disk I/O operation rate within the current sample interval (I/O operations per second)
`qlen`	Number of I/O requests in wait state within the current sample interval
`wait`	Average wait time per I/O within the current sample interval (msec)
`type`	If applicable, identifies what the device is used for. Possible values are `SWAP`, `SYS`, `OCR`, `ASM`, and `VOTING`.

Table J–6 NICS View Metric Descriptions

Metric	Description
`netrr`	Average network receive rate within the current sample interval (KB per second)
`netwr`	Average network sent rate within the current sample interval (KB per second)
`neteff`	Average effective bandwidth within the current sample interval (KB per second)
`nicerrors`	Average error rate within the current sample interval (errors per second)
`pktsin`	Average incoming packet rate within the current sample interval (packets per second)
`pktsout`	Average outgoing packet rate within the current sample interval (packets per second)
`errsin`	Average error rate for incoming packets within the current sample interval (errors per second)
`errsout`	Average error rate for outgoing packets within the current sample interval (errors per second)

Table J–6 (Cont.) NICS View Metric Descriptions

Metric	Description
`indiscarded`	Average drop rate for incoming packets within the current sample interval (packets per second)
`outdiscarded`	Average drop rate for outgoing packets within the current sample interval (packets per second)
`inunicast`	Average packet receive rate for unicast within the current sample interval (packets per second)
`type`	Whether PUBLIC or PRIVATE
`innonunicast`	Average packet receive rate for multi-cast (packets per second)
`latency`	Estimated latency for this network interface card (msec)

Table J–7 FILESYSTEMS View Metric Descriptions

Metric	Description
`total`	Total amount of space (KB)
`mount`	Mount point
`type`	File system type, whether local file system, NFS, or other
`used`	Amount of used space (KB)
`available`	Amount of available space (KB)
`used%`	Percentage of used space (%)
`ifree%`	Percentage of free file nodes (%) **Note:** This metric is not available on Windows systems.

Table J–8 PROTOCOL ERRORS View Metric Descriptions[1]

Metric	Description
`IPHdrErr`	Number of input datagrams discarded due to errors in their IPv4 headers
`IPAddrErr`	Number of input datagrams discarded because the IPv4 address in their IPv4 header's destination field was not a valid address to be received at this entity
`IPUnkProto`	Number of locally-addressed datagrams received successfully but discarded because of an unknown or unsupported protocol
`IPReasFail`	Number of failures detected by the IPv4 reassembly algorithm
`IPFragFail`	Number of IPv4 discarded datagrams due to fragmentation failures
`TCPFailedConn`	Number of times that TCP connections have made a direct transition to the CLOSED state from either the SYN-SENT state or the SYN-RCVD state, plus the number of times that TCP connections have made a direct transition to the LISTEN state from the SYN-RCVD state
`TCPEstRst`	Number of times that TCP connections have made a direct transition to the CLOSED state from either the ESTABLISHED state or the CLOSE-WAIT state
`TCPRetraSeg`	Total number of TCP segments retransmitted

Table J–8 (Cont.) PROTOCOL ERRORS View Metric Descriptions[1]

Metric	Description
`UDPUnkPort`	Total number of received UDP datagrams for which there was no application at the destination port
`UDPRcvErr`	Number of received UDP datagrams that could not be delivered for reasons other than the lack of an application at the destination port

[1] All protocol errors are cumulative values since system startup.

Table J–9 CPUS View Metric Descriptions

Metric	Description
cpu*id*	Virtual CPU
sys-*usage*	CPU usage in system space
user-*usage*	CPU usage in user space
nice	Value of NIC for a specific CPU
usage	CPU usage for a specific CPU
iowait	CPU wait time for I/O operations

Example J–1 Sample Node View

```
----------------------------------------
Node: rwsak10 Clock: '14-04-16 18.47.25 PST8PDT' SerialNo:155631
----------------------------------------

SYSTEM:
#pcpus: 2 #vcpus: 24 cpuht: Y chipname: Intel(R) cpu: 1.23 cpuq: 0
physmemfree: 8889492 physmemtotal: 74369536 mcache: 55081824 swapfree: 18480404
swaptotal: 18480408 hugepagetotal: 0 hugepagefree: 0 hugepagesize: 2048 ior: 132
iow: 236 ios: 23 swpin: 0 swpout: 0 pgin: 131 pgout: 235 netr: 72.404
netw: 97.511 procs: 969 procsoncpu: 6 rtprocs: 62 rtprocsoncpu N/A #fds: 32640
#sysfdlimit: 6815744 #disks: 9 #nics: 5 nicErrors: 0

TOP CONSUMERS:
topcpu: 'osysmond.bin(30981) 2.40' topprivmem: 'oraagent.bin(14599) 682496'
topshm: 'ora_dbw2_oss_3(7049) 2156136' topfd: 'ocssd.bin(29986) 274'
topthread: 'java(32255) 53'

CPUS:

cpu18: sys-2.93 user-2.15 nice-0.0 usage-5.8 iowait-0.0 steal-0.0
.
.
.

PROCESSES:

name: 'osysmond.bin' pid: 30891 #procfdlimit: 65536 cpuusage: 2.40 privmem: 35808
shm: 81964 #fd: 119 #threads: 13 priority: -100 nice: 0 state: S
.
.
.

DEVICES:
```

```
sdi ior: 0.000 iow: 0.000 ios: 0 qlen: 0 wait: 0 type: SYS
sda1 ior: 0.000 iow: 61.495 ios: 629 qlen: 0 wait: 0 type: SYS
.
.
.

NICS:

lo netrr: 39.935  netwr: 39.935  neteff: 79.869  nicerrors: 0 pktsin: 25
pktsout: 25  errsin: 0  errsout: 0  indiscarded: 0  outdiscarded: 0
inunicast: 25 innonunicast: 0  type: PUBLIC
eth0 netrr: 1.412  netwr: 0.527  neteff: 1.939  nicerrors: 0 pktsin: 15
pktsout: 4  errsin: 0  errsout: 0  indiscarded: 0  outdiscarded: 0
inunicast: 15  innonunicast: 0  type: PUBLIC  latency: <1

FILESYSTEMS:

mount: / type: rootfs total: 563657948 used: 78592012 available: 455971824
used%: 14 ifree%: 99 GRID_HOME
.
.
.

PROTOCOL ERRORS:

IPHdrErr: 0 IPAddrErr: 0 IPUnkProto: 0 IPReasFail: 0 IPFragFail: 0
TCPFailedConn: 5197 TCPEstRst: 717163 TCPRetraSeg: 592 UDPUnkPort: 103306
UDPRcvErr: 70
```

oclumon manage

Use the `oclumon manage` command to view and change configuration information from the system monitor service.

Syntax

```
oclumon manage -repos {{changeretentiontime time} | {changerepossize
memory_size}} | -get {key1 [key2 ...] | alllogger [-details] | mylogger
[-details]}
```

Parameters

Table J–10 oclumon manage Command Parameters

Parameter	Description
`-repos {{changeretentiontime time}	{changerepossize memory_size}}`	The `-repos` flag is required to specify the following CHM repository-related options: ■ `changeretentiontime` *`time`*: Use this option to confirm that there is sufficient tablespace to hold the amount of CHM data that can be accumulated in a specific amount of time. **Note:** This option *does not* change retention time. ■ `changerepossize` *`memory_size`*: Use this option to change the CHM repository space limit to a specified number of MB **Caution:** If you decrease the space limit of the CHM repository, then all data collected before the resizing operation is permanently deleted.

Table J–10 (Cont.) oclumon manage Command Parameters

Parameter	Description
`-get` *`key1`* `[`*`key2`* `...]`	Use this option to obtain CHM repository information using the following keywords: `repsize`: Size of the CHM repository, in seconds `reppath`: Directory path to the CHM repository `master`: Name of the master node `alllogger`: Special key to obtain a list of all nodes running Cluster Logger Service `mylogger`: Special key to obtain the node running the Cluster Logger Service which is serving the current node ■ `-details`: Use this option with `alllogger` and `mylogger` for listing nodes served by the Cluster Logger Service You can specify any number of keywords in a space-delimited list following the `-get` flag.
`-h`	Displays online help for the `oclumon manage` command

Usage Notes

- The local system monitor service must be running to change the retention time of the CHM repository.
- The Cluster Logger Service must be running to change the retention time of the CHM repository.

Examples

The following examples show commands and sample output:

```
$ oclumon manage -get MASTER
Master = node1

$ oclumon manage -get alllogger -details
Logger = node1
Nodes = node1,node2

$ oclumon manage -repos changeretentiontime 86400

$ oclumon manage -repos changerepossize 6000
```

oclumon version

Use the `oclumon version` command to obtain the version of CHM that you are using.

Syntax

```
oclumon version
```

Example

This command produces output similar to the following:

```
Cluster Health Monitor (OS), Version 12.1.0.0.2 - Production Copyright 2013
Oracle. All rights reserved.
```

Oracle Clusterware Diagnostic and Alert Log Data

Oracle Clusterware uses Oracle Database fault diagnosability infrastructure to manage diagnostic data and its alert log. As a result, most diagnostic data resides in the

Automatic Diagnostic Repository (ADR), a collection of directories and files located under a base directory that you specify during installation. This section describes clusterware-specific aspects of how Oracle Clusterware uses ADR.

See Also:

- *Oracle Database Administrator's Guide* for more information on the fault diagnosability infrastructure and ADR
- *Oracle Database Utilities* for information about the ADR Command Interpreter (ADRCI)

This section includes the following topics:

- ADR Directory Structure
- Files in the Trace Directory
- The Oracle Clusterware Alert Log
- Incident Trace Files
- Other Diagnostic Data

ADR Directory Structure

Oracle Clusterware ADR data is written under a root directory known as the ADR base. Because components other than ADR use this directory, it may also be referred to as the Oracle base. You specify the file system path to use as the base during Oracle Grid Infrastructure installation and can only be changed if you reinstall the Oracle Grid Infrastructure.

ADR files reside in an ADR home directory. The ADR home for Oracle Clusterware running on a given host always has this structure:

```
ORACLE_BASE/diag/crs/host_name/crs
```

In the preceding example, `ORACLE_BASE` is the Oracle base path you specified when you installed the Oracle Grid Infrastructure and `host_name` is the name of the host. On a Windows platform, this path uses backslashes (\) to separate directory names.

Under the ADR home are various directories for specific types of ADR data. The directories of greatest interest are `trace` and `incident`. The `trace` directory contains all normal (non-incident) trace files written by Oracle Clusterware daemons and command-line programs as well as the simple text version of the Oracle Clusterware alert log. This organization differs significantly from versions prior to Oracle Clusterware 12*c* release 1 (12.1.0.2), where diagnostic log files were written under distinct directories per daemon.

Files in the Trace Directory

Starting with Oracle Clusterware 12*c* release 1 (12.1.0.2), diagnostic data files written by Oracle Clusterware programs are known as *trace files* and have a `.trc` file extension, and appear together in the `trace` subdirectory of the ADR home. The naming convention for these files generally uses the executable program name as the file name, possibly augmented with other data depending on the type of program.

Trace files written by Oracle Clusterware command-line programs incorporate the Operating System process ID (PID) in the trace file name to distinguish data from multiple invocations of the same command program. For example, trace data written by CRSCTL uses this name structure: `crsctl_PID.trc`. In this example, *`PID`* is the operating system process ID displayed as decimal digits.

Trace files written by Oracle Clusterware daemon programs do not include a PID in the file name, and they also are subject to a file rotation mechanism that affects naming. Rotation means that when the current daemon trace file reaches a certain size, the file is closed, renamed, and a new trace file is opened. This occurs a fixed number of times, and then the oldest trace file from the daemon is discarded, keeping the rotation set at a fixed size.

Most Oracle Clusterware daemons use a file size limit of 10 MB and a rotation set size of 10 files, thus maintaining a total of 100 MB of trace data. The current trace file for a given daemon simply uses the program name as the file name; older files in the rotation append a number to the file name. For example, the trace file currently being written by the Oracle High Availability Services daemon (OHASD) is named `ohasd.trc`; the most recently rotated-out trace file is named `ohasd_n.trc`, where *`n`* is an ever-increasing decimal integer. The file with the highest *`n`* is actually the most recently archived trace, and the file with the lowest *`n`* is the oldest.

Oracle Clusterware agents are daemon programs whose trace files are subject to special naming conventions that indicate the origin of the agent (whether it was spawned by the OHASD or the Cluster Ready Services daemon (CRSD)) and the Operating System user name with which the agent runs. Thus, the name structure for agents is:

```
origin_executable_user_name
```

Note: The first two underscores (_) in the name structure are literal and are included in the trace file name. The underscore in *`user_name`* is not part of the file naming convention.

In the previous example, *`origin`* is either `ohasd` or `crsd`, *`executable`* is the executable program name, and *`user_name`* is the operating system user name. In addition, because they are daemons, agent trace files are subject to the rotation mechanism previously described, so files with an additional `_n` suffix are present after rotation occurs.

The Oracle Clusterware Alert Log

Besides trace files, the `trace` subdirectory in the Oracle Clusterware ADR home contains the simple text Oracle Clusterware alert log. It always has the name `alert.log`. The alert log is also written as an XML file in the `alert` subdirectory of the ADR home, but the text alert log is most easily read.

The alert log is the first place to look when a problem or issue arises with Oracle Clusterware. Unlike the Oracle Database instance alert log, messages in the Oracle Clusterware alert log are identified, documented, and localized (translated). Oracle Clusterware alert messages are written for most significant events or errors that occur.

Note: Messages and data written to Oracle Clusterware trace files generally are not documented and translated and are used mainly by My Oracle Support for problem diagnosis.

Incident Trace Files

Certain errors occur in Oracle Clusterware programs that will raise an ADR incident. In most cases, these errors should be reported to My Oracle Support for diagnosis. The occurrence of an incident normally produces one or more descriptive messages in the Oracle Clusterware alert log.

In addition to alert messages, incidents also cause the affected program to produce a special, separate trace file containing diagnostic data related to the error. These incident-specific trace files are collected in the `incident` subdirectory of the ADR home rather than the `trace` subdirectory. Both the normal trace files and incident trace files are collected and submitted to Oracle when reporting the error.

> **See Also:** *Oracle Database Administrator's Guide* for more information on incidents and data collection

Other Diagnostic Data

Besides ADR data, Oracle Clusterware collects or uses other data related to problem diagnosis. Starting with Oracle Clusterware 12*c* release 1 (12.1.0.2), this data resides under the same base path used by ADR, but in a separate directory structure with this form: *`ORACLE_BASE`*`/crsdata/`*`host_name`*. In this example, *`ORACLE_BASE`* is the Oracle base path you specified when you installed the Grid Infrastructure and *`host_name`* is the name of the host.

In this directory, on a given host, are several subdirectories. The two subdirectories of greatest interest if a problem occurs are named `core` and `output`. The `core` directory is where Oracle Clusterware daemon core files are written when the normal ADR location used for core files is not available (for example, before ADR services are initialized in a program). The `output` directory is where Oracle Clusterware daemons redirect their C standard output and standard error files. These files generally use a name structure consisting of the executable name with the characters *OUT* appended to a `.trc` file extension (like trace files). For example, the redirected standard output from the Cluster Time Synchronization Service daemon is named `octssdOUT.trc`. Typically, daemons write very little to these files, but in certain failure scenarios important data may be written there.

Diagnostics Collection Script

When an Oracle Clusterware error occurs, run the `diagcollection.pl` diagnostics collection script to collect diagnostic information from Oracle Clusterware into trace files. The diagnostics provide additional information so My Oracle Support can resolve problems. Run this script as `root` from the *`Grid_home`*`/bin` directory.

Syntax

Use the diagcollection.pl script with the following syntax:

```
diagcollection.pl {--collect [--crs | --acfs | -all] [--chmos [--incidenttime time
[--incidentduration time]]]
   [--adr location [--aftertime time [--beforetime time]]]
   [--crshome path | --clean | --coreanalyze}]
```

> **Note:** The `diagcollection.pl` script arguments are all preceded by two dashes (`--`).

Parameters

Table J–11 lists and describes the parameters used with the `diagcollection.pl` script.

Table J–11 diagcollection.pl Script Parameters

Parameter	Description
`--collect`	Use this parameter with any of the following arguments: ■ `--crs`: Use this argument to collect Oracle Clusterware diagnostic information ■ `--acfs`: Use this argument to collect Oracle ACFS diagnostic information **Note:** You can only use this argument on UNIX systems. ■ `--all`: Use this argument to collect all diagnostic information except CHM (OS) data **Note:** This is the default. ■ `--chmos`: Use this argument to collect the following CHM diagnostic information `--incidenttime` *`time`*: Use this argument to collect CHM (OS) data from the specified time **Note:** The time format is MM/DD/YYYYHH24:MM:SS. `--incidentduration` *`time`*: Use this argument with `--incidenttime` to collect CHM (OS) data for the duration after the specified time **Note:** The time format is HH:MM. If you do not use `--incidentduration`, then all CHM (OS) data after the time you specify in `--incidenttime` is collected. ■ `--adr` *`location`*: The Automatic Diagnostic Repository Command Interpreter (ADRCI) uses this argument to specify a location in which to collect diagnostic information for ADR **See Also:** *Oracle Database Utilities* for more information about ADRCI ■ `--aftertime` *`time`*: Use this argument with the `--adr` argument to collect archives after the specified time **Note:** The time format is YYYYMMDDHHMISS24. ■ `--beforetime` *`time`*: Use this argument with the `--adr` argument to collect archives before the specified time **Note:** The time format is YYYYMMDDHHMISS24. ■ `--crshome` *`path`*: Use this argument to override the location of the Oracle Clusterware home **Note:** The `diagcollection.pl` script typically derives the location of the Oracle Clusterware home from the system configuration (either the `olr.loc` file or the Windows registry), so this argument is not required.
`--clean`	Use this parameter to clean up the diagnostic information gathered by the `diagcollection.pl` script. **Note:** You cannot use this parameter with `--collect`.
`--coreanalyze`	Use this parameter to extract information from core files and store it in a text file. **Note:** You can only use this parameter on UNIX systems.

Rolling Upgrade and Driver Installation Issues

During an upgrade, while running the Oracle Clusterware `root.sh` script, you may see the following messages:

- ACFS-9427 Failed to unload ADVM/ACFS drivers. A system restart is recommended.
- ACFS-9428 Failed to load ADVM/ACFS drivers. A system restart is recommended.

If you see these error messages during the upgrade of the initial (first) node, then do the following:

1. Complete the upgrade of all other nodes in the cluster.
2. Restart the initial node.
3. Run the `root.sh` script on initial node again.
4. Run the `*Oracle_home*/cfgtoollogs/configToolAllCommands` script as `root` to complete the upgrade.

For nodes other than the initial node (the node on which you started the installation):

1. Restart the node where the error occurs.
2. Run the `orainstRoot.sh` script as `root` on the node where the error occurs.
3. Change directory to the Grid home, and run the `root.sh` script on the node where the error occurs.

Testing Zone Delegation

> **See Also:** Appendix E, "Oracle Clusterware Control (CRSCTL) Utility Reference" for information about using the CRSCTL commands referred to in this procedure

Use the following procedure to test zone delegation:

1. Start the GNS VIP by running the following command as `root`:

   ```
   # crsctl start ip -A IP_name/netmask/interface_name
   ```

   The *`interface_name`* should be the public interface and netmask of the public network.

2. Start the test DNS server on the GNS VIP by running the following command (you must run this command as `root` if the port number is less than 1024):

   ```
   # crsctl start testdns -address address [-port port]
   ```

   This command starts the test DNS server to listen for DNS forwarded packets at the specified IP and port.

3. Ensure that the GNS VIP is reachable from other nodes by running the following command as `root`:

   ```
   crsctl status ip -A IP_name
   ```

4. Query the DNS server directly by running the following command:

   ```
   crsctl query dns -name name -dnsserver DNS_server_address
   ```

   This command fails with the following error:

   CRS-10023: Domain name look up for name asdf.foo.com failed. Operating system error: Host name lookup failure

Look at *Grid_home*/log/*host_name*/client/odnsd_*.log to see if the query was received by the test DNS server. This validates that the DNS queries are not being blocked by a firewall.

5. Query the DNS delegation of GNS domain queries by running the following command:

```
crsctl query dns -name name
```

Note: The only difference between this step and the previous step is that you are not giving the -dnsserver *DNS_server_address* option. This causes the command to query name servers configured in /etc/resolv.conf. As in the previous step, the command fails with same error. Again, look at odnsd*.log to ensure that odnsd received the queries. If step 5 succeeds but step 6 does not, then you must check the DNS configuration.

6. Stop the test DNS server by running the following command:

```
crsctl stop testdns -address address
```

7. Stop the GNS VIP by running the following command as root:

```
crsctl stop ip -A IP_name/netmask/interface_name
```

Oracle Trace File Analyzer Collector

Note: This feature is not supported on Windows operating systems.

The Oracle Trace File Analyzer (TFA) Collector is a tool for targeted diagnostic collection that simplifies diagnostic data collection for Oracle Clusterware, Oracle Grid Infrastructure, and Oracle RAC systems. TFA collects and packages diagnostic data and also has the ability to centralize and automate the collection of diagnostic information.

TFA is installed into the Oracle Grid Infrastructure home when you install, or upgrade to, Oracle Database 12*c* release 1 (12.1.0.2). The TFA daemon discovers relevant trace file directories and then analyzes the trace files in those directories to determine their file type (whether they are database trace or log files, or Oracle Clusterware trace or log files, for example) and the first and last timestamps of those files. TFA stores this data in a Berkeley database in the Grid home owner's ORACLE_BASE directory.

The TFA daemon periodically checks for new trace directories to add, such as when a new database instance is created, and also periodically checks whether the trace files metadata need updating. TFA uses this metadata when performing diagnostic data collections.

TFA can perform diagnostic collections in two ways, either on demand or automatically:

- Use tfactl set to enable automatic collection of diagnostics upon discovery of a limited set of errors in trace files

  See Also: "tfactl set" on page J-24

- The `tfactl diagcollect` command collects trimmed trace files for any component and specific time range

This section includes the following topics:

- Managing the TFA Daemon
- TFA Control Utility Command Reference
- Data Redaction with TFA

Managing the TFA Daemon

TFA starts automatically whenever a node starts. You can manually start or stop TFA using the following commands:

- `/etc/init.d/init.tfa start`: Starts the TFA daemon
- `/etc/init.d/init.tfa stop`: Stops the TFA daemon
- `/etc/init.d/init.tfa restart`: Stops and then starts the TFA daemon
- `/etc/init.d/init.tfa shutdown`: Stops the TFA daemon and removes entries from the appropriate operating system configuration

If the TFA daemon fails, then the operating system restarts it, automatically.

TFA Control Utility Command Reference

The TFA control utility, TFACTL, is the command-line interface for TFA, and is located in the *`Grid_home`*`/tfa/bin` directory. You must run TFACTL as `root` or `sudo`, because that gives access to trace files that normally allow only root access. Some commands, such as `tfactl host add`, require strict root access.

You can append the `-help` flag to any of the TFACTL commands to obtain online usage information.

This section lists and describes the following TFACTL commands:

- `tfactl print`
- `tfactl purge`
- `tfactl directory`
- `tfactl host`
- `tfactl set`
- `tfactl diagcollect`

tfactl print

Use the `tfactl print` command to print information from the Berkeley database.

Syntax

```
tfactl print [status | config | directories | hosts | actions | repository |
cookie
```

Parameters

Table J–12 tfactl print Command Parameters

Parameter	Description
`status`	Prints the status of TFA across all nodes in the cluster, and also prints the TFA version and the port on which it is running.
`config`	Prints the current TFA configuration settings.
`directories`	Lists all the directories that TFA scans for trace or log file data, and shows the location of the trace directories allocated for the database, Oracle ASM, and instance.
`hosts`	Lists the hosts that are part of the TFA cluster, and that can receive clusterwide commands.
`actions`	Lists all the actions submitted to TFA, such as diagnostic collection. By default, `tfactl print` actions only shows actions that are running or that have completed in the last hour.
`repository`	Prints the current location and amount of used space of the repository directory. Initially, the maximum size of the repository directory is the smaller of either 10GB or 50% of available file system space. If the maximum size is exceeded or the file system space gets to 1GB or less, then TFA suspends operations and the repository is closed. Use the `tfactl purge` command to clear collections from the repository.
`cookie`	Generates and prints an identification code for use by the `tfactl set` command.

Example

The `tfactl print config` command returns output similar to the following:

```
Configuration parameter                                    Value
------------------------------------------------------------
TFA Version                                              2.5.1.5
Automatic diagnostic collection                              OFF
Trimming of files during diagcollection                       ON
Repository current size (MB) in node1                        526
Repository maximum size (MB) in node1                      10240
Trace Level                                                    1
```

In the preceding sample output:

- **`Automatic diagnostic collection`**: When `ON` (default is `OFF`), if scanning an alert log, then finding specific events in those logs triggers diagnostic collection.
- **`Trimming of files during diagcollection`**: Determines if TFA will trim large files to contain only data that is within specified time ranges. When this is `OFF`, no trimming of trace files occurs for automatic diagnostic collection.
- **`Repository current size in MB`**: How much space in the repository is currently used.
- **`Repository maximum size in MB`**: The maximum size of storage space in the repository. Initially, the maximum size is set to the smaller of either 10GB or 50% of free space in the file system.
- **`Trace Level`**: 1 is the default, and the values 2, 3, and 4 have increasing verbosity. While you can set the trace level dynamically for the running TFA daemon, increasing the trace level significantly impacts the performance of TFA, and should only be done at the request of My Oracle Support.

tfactl purge

Use the `tfactl purge` command to delete diagnostic collections from the TFA repository that are older than a specific time.

Syntax

```
tfactl purge -older number[h | d]
```

Specify a number followed by `h` or `d` (to specify a number of hours or days, respectively) to remove files older than the specified time constraint. For example:

```
# tfactl purge -older 30d
```

The preceding command removes files older than 30 days.

tfactl directory

Use the `tfactl directory` command to add a directory to, or remove a directory from, the list of directories that will have their trace or log files analyzed. You can also use this command to change the directory permissions. When a directory is added by auto discovery, it is added as public, which means that any file in that directory can be collected by any user that has permission to run the `tfactl diagcollect` command. This is only important when `sudo` is used to allow users other than root to run TFACTL commands. If a directory is marked as private, then TFA determines which user is running TFACTL commands and verifies that the user has permissions to see the files in the directory before allowing any files to be collected.

> **Note:** A user can only add a directory to TFA to which they have read access, and also that TFA auto collections, when configured, run as `root`, and so will always collect all available files.

Syntax

```
tfactl directory [add directory | remove directory | modify directory -private |
-public]
```

Parameters

Table J–13 tfactl directory Command Parameters

Parameter	Description
`add` *`directory`*	Adds a specific directory
`remove` *`directory`*	Removes a specific directory
`modify` *`directory`* `-private | -public`	Modifies a specific directory to either be private or public, where information can only be collected by users with specific operating system permissions (`-private`), or by anyone with the ability to run the `tfactl diagcollection` command (`-public`)

Usage Notes

You must add all trace directory names to the Berkeley database, so that TFA will collect file metadata in that directory. The discovery process finds most directories but if new or undiscovered directories are required, then you can add these manually using the `tfactl directory` command. When you add a directory using TFACTL, TFA attempts to determine whether the directory is for the database, Oracle Clusterware, operating system logs, or some other component, and for which database or instance. If TFA cannot determine this information, then it returns an error and requests that you enter the information, similar to the following:

```
# tfactl directory add /tmp

Failed to add directory to TFA. Unable to determine parameters for directory: /tmp
Please enter component for this Directory [RDBMS|CRS|ASM|INSTALL|OS|CFGTOOLS] :
RDBMS
Please enter database name for this Directory :MYDB
Please enter instance name for this Directory :MYDB1
```

tfactl host

Use the `tfactl host` command to add hosts to, or remove hosts from, the TFA cluster.

Syntax

```
tfactl host [add host_name | remove host_name]
```

Specify a host name to add or remove, as in the following example:

```
# tfactl host add myhost.domain.com
```

Usage Notes

Using the `tfactl host` add command notifies the local TFA about other nodes on your network. When you add a host, TFA contacts that host and, if TFA is running on that host, then both hosts synchronize their host list. TFA authenticates that a host can be added using a cookie. If the host to be added does not have the correct cookie, then you must retrieve that cookie from an existing host in the cluster and set it on the host being added, similar to the following:

```
#tfactl host add node2

Failed to add host: node2 as the TFA cookies do not match.
To add the host successfully, try the following steps:
1. Get the cookie in node1 using:
  ./tfa_home/bin/tfactl print cookie
2. Set the cookie from Step 1 in node2 using:
  ./tfa_home/bin/tfactl set cookie=<COOKIE>
3. After Step 2, add host again:
  ./tfa_home/bin/tfactl host add node2
```

After you successfully add the host, all clusterwide commands will activate on all nodes registered in the Berkeley database.

tfactl set

Use the `tfactl set` command to adjust the manner in which TFA runs.

Syntax

```
tfactl set [autodiagcollect=ON | OFF | cookie=UID | trimfiles=ON | OFF |
tracelevel=1 | 2 | 3 | 4 | reposizeMB=number | repositorydir=directory] [-c]
```

Parameters

Table J–14 tfactl set Command Parameters

Parameter	Description
`autodiagcollect=ON | OFF`	When set to `OFF`, which is the default, automatic diagnostic collection is disabled. If set to `ON`, TFA automatically collects diagnostics when certain patterns occur while TFA scans the alert logs. To set automatic collection for all nodes of the TFA cluster, you must specify the `-c` parameter.
`cookie=`*`UID`*	Use the `tfactl print cookie` command to generate the cookie identification.
`trimfiles=ON | OFF`	When set to `ON`, TFA trims files to only have relevant data when diagnostic collection is done as part of a scan. Note: When using `tfactl diagcollect`, you determine the time range for trimming with the parameters you specify with that command. Oracle recommends that you *not* set this parameter to `OFF`, because untrimmed data be very large.
`tracelevel=1 | 2 | 3 | 4`	Do not change the tracing level unless you are directed to do so by My Oracle Support.
`-c`	Specify this parameter to propagate these settings to all nodes in the TFA configuration.

Automatic Diagnostic Collection

After TFA initially gathers trace file metadata, the daemon monitors all files that are determined to be alert logs using `tail`, so that TFA can take action when certain strings occur.

By default, these logs are database alert logs, Oracle ASM alert logs, and Oracle Clusterware alert logs. When specific patterns occur in the logs saved to the Berkeley database, automatic diagnostic collection may take place.

Exactly what is collected depends on the pattern that is matched. TFA may just store information on the pattern matched or may initiate local diagnostic collection. Although TFA always monitors the logs and collects information into its database, automatic diagnostic collection only happens if it is enabled first using the `tfactl set` command.

tfactl diagcollect

Use the `tfactl diagcollect` command to perform on-demand diagnostic collection. You can configure a number of different parameters to determine how large or detailed a collection you want. You can specify a specific time of an incident or a time range for data to be collected, and determine if whole files that have relevant data should be collected or just a time interval of data from those files.

Note: If you specify no parameters, then the `tfactl diagcollect` command collects files from all nodes for all components where the file has been updated in the last four hours, and also trims excessive files. If an incident occurred prior to this period, then you can use the parameters documented in this section to target the correct data collection period.

Syntax

```
tfactl diagcollect [-all | -database all | database_1,database_2,... | -asm | -crs
| -os | -install]
[-node all | local | node_1,node_2,...][-tag description] [-z file_name]
[-since numberh | d | -from "mmm/dd/yyyy hh:mm:ss" -to "mmm/dd/yyyy hh:mm:ss"
| -for "mmm/dd/yyyy hh:mm:ss" [-nocopy] [-nomonitor]]
```

Parameters

Table J–15 tfactl diagcollect Command Parameters

Parameter	Description
`-all | -database all | database_1,database_2,... | -asm | -crs | -os | -install`	You can choose one or more individual components from which to collect trace and log files or choose `-all` to collect all files in the inventory. If you do not specify a time interval, then TFACTL collects files from the last four hours.
`-node all | local | node_1,node_2,...`	You can specify a comma-delimited list of nodes from which to collect diagnostic information. Default is `all`.
`-tag description`	Use this parameter to create a subdirectory for the resulting collection in the TFA repository.
`-z file_name`	Use this parameter to specify an output file name.
`-since numberh | d | -from "mmm/dd/yyyy hh:mm:ss" -to "mmm/dd/yyyy hh:mm:ss" | -for "mmm/dd/yyyy hh:mm:ss"`	Specify the `-since` parameter to collect files that have relevant data for the past specific number of hours (`h`) or days (`d`). By default, using the command with this parameter also trims files that are large and shows files only from the specified interval. Specify the `-from` and `-to` parameters (you must use these two parameters together) to collect files that have relevant data during a specific time interval, and trim data before this time where files are large. Specify the `-for` parameter to collect files that have relevant data for the time given. The files TFACTL collects will have timestamps in between which the time you specify after `-for` is included. No data trimming is done for this option. **Note:** If you specify both date and time, then you must enclose both values in double quotation marks (`""`). If you specify only the date or the time, then you do not have to enclose the single value in quotation marks.
`-nocopy`	Specify this parameter to stop the resultant trace file collection from being copied back to the initiating node. The file remains in the TFA repository on the executing node.
`-nomonitor`	Specify this parameter to prevent the terminal on which you run the command from displaying the progress of the command.

Examples

- The following command trims and zips all files updated in the last four hours, including `chmos` and `osw` data, from across the cluster and collects it on the initiating node:

  ```
  # tfactl diagcollect
  ```

- The following command trims and zips all files updated in the last eight hours, including `chmos` and `osw` data, from across the cluster and collects it on the initiating node:

  ```
  # tfactl diagcollect -all -since 8h
  ```

- The following command trims and zips all files from databases hrdb and fdb updated in the last one day and collects it on the initiating node:

  ```
  # tfactl diagcollect -database hrdb,fdb -since 1d -z foo
  ```

- The following command trims and zips all Oracle Clusterware files, operating system logs, and chmos and osw data from node1 and node2 updated in the last six hours, and collects it on the initiating node:

  ```
  # tfactl diagcollect -crs -os -node node1,node2 -since 6h
  ```

- The following command trims and zips all Oracle ASM logs from node1 updated between July 4, 2014 and July 5, 2015 at 21:00, and collects it on the initiating node:

  ```
  # tfactl diagcollect -asm -node node1 -from Jul/4/2014 -to "Jul/5/2014
  21:00:00"
  ```

- The following command trims and zips all log files updated on July 2, 2014 and collects it on the initiating node:

  ```
  # tfactl diagcollect -for Jul/2/2014
  ```

- The following command trims and zips all log files updated from 09:00 on July 2, 2014, to 09:00 on July 3, 2014, which is 12 hours before and after the time specified in the command, and collects it on the initiating node:

  ```
  # tfactl diagcollect -for "Jul/2/2014 21:00:00"
  ```

Data Redaction with TFA

You can hide sensitive data by replacing data in files or paths in file names. To use this feature, place a file called redaction.xml in the *tfa_home*/resources directory. TFA uses the data contained within the redaction.xml file to replace strings in file names and contents. The format of the redaction.xml file is as follows:

```
<replacements>
<replace>
<before>securestring</before>
<after>nonsecurestring</after>
</replace>
<replace>
<before>securestring2</before>
<after>nonsecurestring2</after>
</replace>
Etc…
</replacements>
```

Oracle Clusterware Alerts

Oracle Clusterware writes messages to the ADR alert log file (as previously described) for various important events. Alert log messages generally are localized (translated) and carry a message identifier that can be used to look up additional information about the message. The alert log is the first place to look if there appears to be problems with Oracle Clusterware.

The following is an example of alert log messages from two different CRS daemon processes:

```
2014-07-16 00:27:22.074 [CTSSD(12817)]CRS-2403:The Cluster Time Synchronization
Service on host stnsp014 is in observer mode.
```

```
2014-07-16 00:27:22.146 [CTSSD(12817)]CRS-2407:The new Cluster Time
Synchronization Service reference node is host stnsp013.
2014-07-16 00:27:22.753 [CTSSD(12817)]CRS-2401:The Cluster Time Synchronization
Service started on host stnsp014.
2014-07-16 00:27:43.754 [CRSD(12975)]CRS-1012:The OCR service started on node
stnsp014.
2014-07-16 00:27:46.339 [CRSD(12975)]CRS-1201:CRSD started on node stnsp014.
```

The following example shows the start of the Oracle Cluster Time Synchronization Service (OCTSS) after a cluster reconfiguration:

```
2014-07-15 23:51:17.532 [CTSSD(12813)]CRS-2403:The Cluster Time Synchronization
Service on host stnsp014 is in observer mode.
2014-07-15 23:51:18.292 [CTSSD(12813)]CRS-2407:The new Cluster Time
Synchronization Service reference node is host stnsp013.
2014-07-15 23:51:18.961 [CTSSD(12813)]CRS-2401:The Cluster Time Synchronization
Service started on host stnsp014.
```

Alert Messages Using Diagnostic Record Unique IDs

Beginning with Oracle Database 11*g* release 2 (11.2), certain Oracle Clusterware messages contain a text identifier surrounded by "`(:`" and "`:)`". Usually, the identifier is part of the message text that begins with "`Details in...`" and includes an Oracle Clusterware diagnostic log file path and name similar to the following example. The identifier is called a DRUID, or diagnostic record unique ID:

```
2014-07-16 00:18:44.472 [ORAROOTAGENT(13098)]CRS-5822:Agent
 '/scratch/12.1/grid/bin/orarootagent_root' disconnected from server. Details at
 (:CRSAGF00117:) in /scratch/12.1/grid/log/stnsp014/agent/crsd/orarootagent_
root/orarootagent_root.log.
```

DRUIDs are used to relate external product messages to entries in a diagnostic log file and to internal Oracle Clusterware program code locations. They are not directly meaningful to customers and are used primarily by My Oracle Support when diagnosing problems.

Glossary

action script

A script that defines the start, stop and check actions for a resource. The start action is invoked while starting the resource, the stop action for stopping the resource, and the check action while checking the running status of a resource.

administrator managed

Database administrators define on which servers a database resource should run, and place resources manually as needed. This is the management strategy used in previous releases.

agent

A program that contains the agent framework and user code to manage resources.

agent framework

A C library that enables users to plug in application-specific code to manage customized applications.

availability directive

Instructions to Oracle Clusterware to reconfigure the system when a server leaves or joins a cluster.

cardinality

The number of servers on which a **resource** can run, simultaneously.

client cluster

In a multicluster environment, a client cluster advertises its names with the **server cluster**.

cluster

Multiple interconnected computers or servers that appear as if they are one server to end users and applications.

cluster-aware

Any application designed to be deployed using clusterware.

cluster administrator

An administrator who can manage a certain part of a cluster based on set policies and privileges.

cluster configuration policy

A document for policy-managed clusters, which contains exactly one definition for each server pool defined in the system.

cluster configuration policy set

A document that defines the names of all server pools configured in the cluster, which contains one or more configuration policies. Only one policy can be in effect at any one time, but administrators can set different policies to be in effect at different dates or times of the day in accordance with business needs and system demands.

Cluster Health Monitor

Detects and analyzes operating system and cluster resource-related degradation and failures.

cluster resource

A resource that is aware of the cluster environment and subject to cross-node switchover and failover.

Cluster Time Synchronization Service

A time synchronization mechanism that ensures that all internal clocks of all nodes in a cluster are synchronized.

Cluster Verification Utility (CVU)

A tool that verifies a wide range of Oracle RAC components such as shared storage devices, networking configurations, system requirements, Oracle Clusterware, groups, and users.

database cloud

A set of databases integrated by the global service and load management framework into a single virtual server that offers one or more global services, while ensuring high performance, availability, and optimal utilization of resources.

dependency

The relationship between two or more resources. and the interaction expressed between them.

disk group

An Oracle ASM disk group is a collection of disks that Oracle ASM manages as a unit. Within a disk group, Oracle ASM exposes a file system interface for Oracle Database files. The content of files that is stored in a disk group is evenly distributed, or striped, to eliminate hot spots and to provide uniform performance across the disks. Oracle ASM files may also be optionally mirrored within a disk group. The performance of disks in a disk group is comparable to the performance of raw devices.

Dynamic Host Configuration Protocol (DHCP)

A network application protocol used by devices (DHCP clients) to obtain configuration information for operation in an Internet Protocol network. This protocol reduces system administration workload, allowing devices to be added to the network with little or no manual intervention.

fixup script

Oracle Universal Installer detects when minimum requirements for installation are not completed, and creates shell script programs, called fixup scripts, to resolve many incomplete system configuration requirements. If Oracle Universal Installer detects an incomplete task, it prompts you to create a fixup script and then to run the fixup script in a separate terminal session. You can also generate fixup scripts with certain CVU commands by using the `-fixup` flag.

gold image

A copy of a software only, installed Oracle home, where the home-specific details are unavailable. This enables you to easily copy an image of an Oracle home to a new host on a new file system to serve as an active usable Oracle home.

A gold image can correspond, generally, to any application but in the context of this document, gold images correspond to Oracle Databases. Also, a gold image is *not* a ship home, which means that you do not have to run Oracle Universal Installer when deploying a gold image.

Grid Home client

The Grid Home client is a remote **Oracle Grid Infrastructure** cluster that subscribes to the **Rapid Home Provisioning Server** for provisioning of **gold image**s.

Oracle Grid Infrastructure

The software that provides the infrastructure for an enterprise grid architecture. In a cluster this software includes Oracle Clusterware and Oracle ASM. For a standalone server, this software includes Oracle Restart and Oracle ASM. Oracle Database 12*c* combines these infrastructure products into one software installation called the *Oracle Grid Infrastructure home* (`Grid_home`).

Grid Naming Service (GNS)

A generic service which resolves the names of hosts in a delegated normal DNS zone by mapping them to IP addresses within the zone. GNS enables the use of Dynamic Host Configuration Protocol (DHCP) address for Oracle RAC database nodes, simplifying deployment. GNS also resolves host names passed back from a SCAN listener.

Hub anchor

A Hub Node in an Oracle Flex Cluster that acts as the connection point for purposes of cluster membership for one or more Leaf Nodes. Leaf Nodes exchange heartbeats with a single Hub anchor.

Hub Node

A node in and **Oracle Flex Cluster** that is tightly connected with other servers and has direct access to a shared disk.

image

Master copy of Oracle database home software which resides on the Rapid Home Provisioning Server.

IPv4

Internet Protocol Version 4. IPv4 is the current standard for the IP protocol. IPv4 uses 32-bit (four-byte) addresses, which are typically represented in dotted-decimal notation. The decimal value of each octet is separated by a period, as in 192.168.2.22.

IPv6

Internet Protocol Version 6. The protocol designed to replace IPv4. In IPv6, an IP address is typically represented in eight fields of hexadecimal values separated by colons, as in 2001:0DB8:0000:0000:0000:0000:1428:57AB. In some cases, fields with 0 values can be compressed, as in 2001:DB8::1428:57AB.

Leaf Node

Servers that are loosely coupled with **Hub Node**s, which may not have direct access to the shared storage.

local resource

A resource that runs on the nodes of a cluster but is unaware of anything outside of the scope of the node.

OCR writer

Each CRSD process also acts as an OCR server. One of the CRSD processes in the cluster is the OCR server that performs I/O to the disk group or file, or block or raw device.

Oracle Automatic Storage Management (Oracle ASM)

Oracle ASM manages the storage disks used by Oracle Clusterware and Oracle RAC in disk groups. By default, Oracle Clusterware uses Oracle ASM to store OCR and voting files.

Oracle Cluster Registry (OCR)

The Oracle RAC configuration information repository that manages information about the cluster node list and instance-to-node mapping information. OCR also manages information about Oracle Clusterware resource profiles for customized applications.

Oracle Clusterware

Software that allows groups (clusters) of connected servers to operate or be controlled as a unit.

Oracle Clusterware stack

The Oracle Clusterware stack includes Oracle Cluster Ready Services, Event Manager, Cluster Synchronization Services, and Oracle ASM (if used).

Oracle Flex Cluster

Large clusters that are made of up of two types of nodes: **Hub Node**s and **Leaf Node**s, where the Hub Nodes form a cluster using current membership algorithms and Leaf Nodes connect for membership to a single Hub Node called a **Hub anchor**.

policy managed

Database administrators specify the server pool (excluding Generic or Free) in which the database resource runs. Oracle Clusterware places the database resource on a server.

Rapid Home Provisioning Server

The Rapid Home Provisioning Server is one server on the server cluster that can hold a repository of **gold image**s.

resource

A database, application, or process managed by Oracle Clusterware.

resource state

The state of a particular resource at any given time that determines its availability to the cluster.

resource type

Defines whether a resource is either a **cluster resource** or a **local resource**.

server categorization

A method of identifying servers in the cluster into different categories by using a set of server attributes. Administrators can configure server pools to restrict which categories they accept. Server categories are created by providing attributes for the SERVER_CATEGORY parameter.

server cluster

In a multicluster environment, a server cluster is a cluster in which the Grid Naming Service (GNS) process runs.

server pool

A logical division of servers in a cluster into a group that hosts applications, databases, or both.

Single Client Access Name (SCAN)

A single name that resolves to three IP addresses in the public network.

start effort evaluation

The process that Oracle Clusterware goes through when it starts a resource. During this process, Oracle Clusterware considers resource dependencies contained in the profile of the resource.

state

A set of values that describes what the condition of a particular resource.

working copy

An Oracle ACFS snapshot of a **gold image** that can either be a software-only installed Oracle home or an instantiated, configured Oracle home.

Index