

## Storage Foundation for Oracle RAC with Oracle 10g RAC

Integrating Oracle 10g CRS with Storage Foundation for Oracle RAC

## TABLE OF CONTENTS

1	Introduction .....	3
1.1	Oracle 10g Real Application Clusters (RAC) .....	3
1.2	Hardware and Network Requirements.....	3
2	Cluster Ready Services (CRS) .....	3
2.1	Cluster Ready Services (CRS) Background Processes .....	3
2.2	Choosing a Storage Option for Oracle CRS AND OCR Files.....	4
2.3	Guidelines for Placing Oracle CRS Files on a File System .....	4
2.4	Creating directories Required for CRS Files.....	4
2.5	Starting CRS and monitoring the voting disks .....	5
2.6	Benefits of integrating CRS with Storage Foundation for Oracle RAC.....	5
3	Integrating Oracle 10g RAC with VERITAS Storage Foundation for Oracle RAC .....	5
3.1	Installing Oracle 10g .....	6
3.2	VERITAS Point Patch .....	6
3.3	Configuring cssd resources for the SFRAC with Oracle 10g.....	7
4	Oracle Cluster Registry (OCR) .....	7
4.1	Restoring the Oracle Cluster Registry .....	7
4.2	Moving the Oracle Cluster Registry .....	8
4.3	Importing Oracle Cluster Registry Content .....	8
5	Overview of SRVCTL for Administering RAC.....	8
5.1	Using Services in Real Application Clusters Environments.....	8
5.2	Guidelines for Using SRVCTL in Real Application Clusters .....	8
5.3	Obtaining Command Line Help for SRVCTL .....	9
5.4	SRVCTL Command Syntax and Options.....	9
5.5	SRVCTL Cluster Database Configuration Tasks.....	9
5.6	SRVCTL General Cluster Database Administration Tasks .....	9
5.7	SRVCTL Node-Level Tasks.....	10
6	Summary: .....	10
	APPENDEX A - SRVCTL commands to control CRS resources.....	11
	APPENDEX B - main.cf utilizing single cssd resource .....	16
	APPENDEX C- main.cf utilizing two cssd resources .....	19

## 1 Introduction

Oracle 10g Real Application Clusters (RAC) uses integrated cluster-ware software Cluster Ready Services (CRS). CRS is a required component for every Oracle 10g RAC instance. This is a new requirement that is applicable only for Oracle 10g. As a result changes have been made to VERITAS Storage Foundation for Oracle RAC (SFRAC) to integrate with CRS and provide more value added benefits to the customer. This paper is intended to be a technical primer on CRS and its integration with VERITAS Storage Foundation for Oracle RAC (SFRAC). This paper also provides relevant information regarding Oracle Cluster Registry (OCR) and the Server Control Utility (SRVCTL) which are required to effectively manage Oracle 10g RAC.

### 1.1 Oracle 10g Real Application Clusters (RAC)

Oracle 10g Real Application Clusters (RAC) is a cluster database with a shared cache architecture that runs on multiple machines, attached through a cluster interconnect and a shared storage subsystem. An Oracle RAC database appears like a single standard Oracle Database to users. The same maintenance tools and practices used for a single Oracle Database can be used on the entire cluster. The key difference however is the ability of RAC to manage the workload, add nodes or relinquish nodes on demand, based on business process requirements. In Oracle 10g Real Application Clusters, customers can use *services* to define application workloads for each application or for major components within complex applications. Users can use these services to control their workload by creating rules to define where and when the service runs.<sup>1</sup>

### 1.2 Hardware and Network Requirements

To run Oracle 10g RAC each cluster node must have:

- **External shared disks:** Used by CRS as the voting disk and the Oracle Cluster Registry (OCR).
- **One IP (Internet Protocol) address for private interconnect:** This IP address must be separate from the public network and it must have the same interface name on every node that is part of your cluster.
- **One Virtual IP address for client connections:** This is in addition to the operating-system managed public host IP address assigned to the node.
- **Redundant switches:** This is a standard configuration for all clusters independent of size.

## 2 Cluster Ready Services (CRS)

In Oracle 10g Real Application Clusters uses integrated cluster-ware software Cluster Ready Services (CRS). CRS manages clustering related functions including membership, group services, global resource management, and high availability for the database. CRS components also interact with SFRAC to coordinate cluster membership information. The OUI (Oracle Universal Installer) installs CRS on each node. The CRS home can be shared by all nodes or be private to each node but VERITAS recommends private CRS home locations to avoid error messages printed by CRS startup scripts. The home that you select for CRS *must* be different from the default Oracle home.

### 2.1 Cluster Ready Services (CRS) Background Processes

CRS consists of the following processes:

- **oproc** -- Process monitor for the cluster. This process is not required while running SFRAC environment.
- **evmd** -- Event manager daemon that starts the `racgevt` process to manage callouts.
- **ocssd** -- Receives node membership from SFRAC modules. Manages cluster node membership and runs as

---

<sup>1</sup> For more information refer to the “Oracle® Database Concepts, 10g Release 1”

`oracle` user; failure of this process results in cluster restart. VERITAS Cluster Server (VCS), integrated within SFRAC, will monitor this process using an Oracle application agent to make sure that resources required for `ocssd` are not off-lined while `ocssd` is running.

- `crsd` -- runs as `root` user and restarts automatically upon failure. Manages Oracle related resources such as Virtual IP address, listener and database instances (if enabled).

## 2.2 Choosing a Storage Option for Oracle CRS AND OCR Files

You can choose one of the following options for the location of the CRS files:

- VERITAS Cluster File System (recommended and integrated with Storage Foundation for Oracle RAC)
- NFS file system on a certified NAS device
- Shared logical volumes
- Raw partitions

**Note:** *It is not necessary to use the same storage option for each type of file.*

## 2.3 Guidelines for Placing Oracle CRS Files on a File System

The Installer does not suggest a default location for the Oracle Cluster Registry (OCR) or the Oracle CRS voting disk. If you choose to create these files on a file system, use the following guidelines when deciding where to place them:

- You must choose a shared file system, for example, a cluster file system on a shared disk or an NFS file system on a certified NAS device.
- It must have at least 100 MB of free disk space for the OCR and 20 MB of free disk space for the CRS voting disk.
- For improved reliability, you should choose a file system on a highly available storage device, for example, a RAID device that implements mirroring.
- If you are placing the Oracle Cluster Ready Services software on a shared file system, you can use the same file system for these files.
- The `oracle` user must have write permissions to create the files in the path you specify.

Oracle Installation and Configuration Guide **suggests** using clustered file system and documents steps for this option only.

## 2.4 Creating directories Required for CRS Files

Create the recommended mount point for the Clustered file system, and set appropriate owner, group and permissions. For example:

```
# mkdir /ora_crs
# chown oracle:oinstall /ora_crs
# chmod 775 /oracrs
```

Mount the `/oracrs` and create two subdirectories:

```
# mkdir OCR VOTE-disk
```

During the above installation process, the voting disk and the **Oracle Cluster Registry (OCR)** will be created. When prompted by Oracle Installer, you will need to provide:

- The name of a file in the OCR directory or raw volume (for example, a file named `ocr_file` in the directory `/ora_crs/OCR`).
- The name of a file in the VOTE-disk directory or raw volume (for example, a file named `vote_file` in the directory

/ora\_crs/VOTE-disk).

## 2.5 Starting CRS and monitoring the voting disks

CRS will be started by the Installer at the end of the installation. It will also be started at the system startup time from the /etc/init.d/init.crs. Typically the CRS file system will not be mounted at this time. CRS will wait until file system is mounted to complete the startup process. While waiting for the file system to be available, a script will keep probing the file system, and reporting error messages until file system becomes available.

Once CRS started, it will monitor the voting disk access from each node. If at any point of time the voting disk is inaccessible, CRS will treat it as a “split brain” condition and reboot the node. If the voting disk is not managed properly, the system could be highly unstable. It is critical to make the voting disk highly available through reliable VERITAS Volume Manager (VxVM) features like mirroring and multipathing, as otherwise it will create a single point of failure for the application.

## 2.6 Benefits of integrating CRS with Storage Foundation for Oracle RAC

Veritas provides several features to enhance CRS functionality:

1. VERITAS Volume Manager (VxVM) and VERITAS File System (VxFS) is used to increase the reliability of Oracle Cluster Registry (OCR) and the voting disk. While the voting disk and OCR can be placed on a raw device or shared file system a shared file system is often the preferred method for better manageability and reliability. If a raw disk is used, it has to be on the volume accessible by both nodes. Even though this can be accomplished by directly accessing the disk connected to both systems, CVM is the preferred solution for several reasons. Mirroring and multipathing is much more reliable, manageable and flexible to the change than the raw disk.
2. PrivNIC agent provides high availability of the Private IP required by Oracle. PrivNIC agent has been available since the initial support of Oracle 10g in Storage Foundation for Oracle RAC 4.0MP1
3. CSSD Application agent controls the dependency between the CRS and the voting disk. It will assure voting disk availability while CRS is running. This feature will become available in Storage Foundation for Oracle RAC 4.1 and is also available as a point patch to Storage Foundation for Oracle RAC 4.0MP1. It will be discussed in more detail in the following section.
4. Oracle database can be controlled by either CRS or by VCS. The advantage of VCS control is in that VCS will also monitor the resources on which instance depends, like storage or IP. Veritas also provides the additional ISV agents for applications that are running on top of Oracle, therefore offering high availability solution for the whole stack. NOTE: You must use either CRS or VCS to control your database resource. Do not use both mechanisms simultaneously.

## 3 Integrating Oracle 10g RAC with VERITAS Storage Foundation for Oracle RAC

To use Oracle10g in an SFRAC environment, do the following steps.<sup>2</sup>

- Copy the SFRAC libraries into place before Oracle CRS installation.
- Create a home directory on each cluster node for \$CRS\_HOME and include the mount in /etc/fstab. The placement of \$CRS\_HOME on a cluster file system is *not supported*.
- Create shared raw volumes or cluster file system directories for the Cluster Ready Services OCR component

---

<sup>2</sup> For more details on each step please refer to the “Storage Foundation for Oracle RAC 4.0 MP1 Release Notes”.

and the VOTE-disk component.

- Configure private IP addresses for the CRS daemons.
- Configure the PrivNIC resource in the CVM service group; this is optional, but highly recommended.
- Configure the OCR and VOTE-disk volumes or directories in the `main.cf` file.
- Verify that configured resources come online after restarting systems.
- Install Oracle10g CRS and binaries and restart the systems.
- Install Oracle10g database binaries.
- Copy libraries to `$ORACLE_HOME`.

### 3.1 Installing Oracle 10g

The Oracle Database 10g installation requires a two-phase process in which you run the Oracle Universal Installer (OUI) twice. The first phase installs Oracle Cluster Ready Services Release 1 and the second phase installs the Oracle Database 10g software with RAC. The Oracle Database 10g installation process provides single system image for RAC installations and patches<sup>3</sup>.

OUI detects that your system has VERITAS cluster-ware; the Cluster Configuration Information page contains pre-defined node information. Otherwise, the OUI displays the Cluster Configuration Information page without pre-defined node information. In order for this to function properly, it is important to copy the SFRAC libraries into place before starting Oracle CRS installation.<sup>4</sup>

On the Voting Disk Information page, enter a complete path and file name for the file in which you want to store the voting disk and click Next. This must be a shared raw device or a shared file system file. The storage size for the OCR should be at least 100MB and the storage size for the voting disk should be at least 20MB. In addition, Oracle recommends that you use a RAID array for storing the OCR and the voting disk to ensure the continuous availability of the partitions.

### 3.2 VERITAS Point Patch

In the initial release of Storage Foundation for Oracle RAC 4.0, few problems were present preventing smooth integration of Oracle RAC 10g with SFRAC 4.0. The issues were identified to be:

- The recommended configuration for the CRS voting disk is to place it under the Cluster File System. By that we created dependency between the CRS and CFS. This means that CFS mount point has to be online while CRS is running. If this mount point is unmounted CRS does not see the voting disk and takes the action to prevent a split brain condition. In other words, it reboots the node. For example, the `hastop -all` command brings CFS offline and the whole cluster needs to be rebooted.
- Oracle's `init.cssd` is written in a way that prevents VERITAS' `clsinfo` script to be executed. This results in problems in configuration where voting disk and OCR are on disks.

These issues have been resolved with the point patch. An application agent to monitor `cssd` process using the scripts supplied through the point patch. This resource should be configured to depend on resources for the voting disk; OCR etc. to assure all necessary components are in place before `cssd` starts and as it is running. It can also be linked with the database resources to assure proper order of onlining/offlining CRS and DB resources.

---

<sup>3</sup> Refer to the *Oracle Real Application Clusters Installation and Configuration Guide* for additional information.

<sup>4</sup> Refer to the "Storage Foundation for Oracle RAC 4.0 MP1 Release Notes" for details on this step.

This relationship can also be handled through CRS. IN THAT CASE, DO NOT INCLUDE DATABASE RESOURCE WITHIN THE VCS.

The point patch contains:

- Bug fix to Oracle's init.cssd. This bug prevents the script from calling VERITAS' clinfo script.
- Support for CSSD resource in any service group. In the previous configuration, VERITAS insisted on CSSD being in CRS group which depended directly on CVM group.
- Support for OCR and Voting on raw disks. Previously we assumed that those are on CVM/CFS. Thus we expected that CSSD will not come up before CVM/CFS is up. That may not be true any more.
- Support for CVM-VVR where all 3 levels of group dependencies are exhausted, thus we cannot have CSSD depending on multiple groups where Oracle data mounts are present. VERITAS now supports multiple resources for CSSD.
- Online, offline, monitor and clean scripts for Application agent to work with ocspd process
- Modified init.cssd and clinfo Scripts to allow execution of the Veritas Cluster check
- Main.cf examples to include the cssd resources.

This point patch will be rolled into the future releases of the SFRAC, starting with the SFRAC 4.1. For more information and to acquire the point patch please contact VERITAS Support.

### 3.3 Configuring cssd resources for the SFRAC with Oracle 10g

Modify main.cf to include one or more Application resource(s) for cssd. Sample main.cf files are included at the end of this document. Multiple cssd resources are only required if resources on which CSSD depends are scattered across multiple groups. The cssd resources \*must\* have the following attributes.

```
Application cssd (
  Critical = 0
  StartProgram = "/opt/VRTSvcs/rac/bin/cssd-online"
  StopProgram = "/opt/VRTSvcs/rac/bin/cssd-offline"
  CleanProgram = "/opt/VRTSvcs/rac/bin/cssd-clean"
  MonitorProgram = "/opt/VRTSvcs/rac/bin/cssd-monitor"
  OnlineRetryLimit = 20
)
```

## 4 Oracle Cluster Registry (OCR)

The Oracle Cluster Registry (OCR) contains cluster and database configuration information for Oracle 10g RAC and CRS. Some of this information includes the cluster node list, cluster database instance-to-node mapping information, and the CRS application resource profiles.

OCR is backed up automatically by Oracle. In addition to using the automatically created OCR backup files, you should also export the OCR contents before and after making significant configuration changes. If you run into irresolvable configuration problems, or if you are unable to restart your cluster-ware, restore your configuration using the procedure described in the next section

### 4.1 Restoring the Oracle Cluster Registry

1. Stop the CRS software on all of the nodes in your cluster database by executing the `init.crs stop` command on all of the nodes.
2. Identify the recent backups using the `ocrconfig -showbackup` command.

3. Execute the restore by applying an OCR backup file identified in Step 2 with the `ocrconfig -restore file name` command.
4. Restart the CRS software on all of the nodes in your cluster by restarting each node.

## 4.2 Moving the Oracle Cluster Registry

1. Stop the CRS software on all of the nodes in your cluster database by executing the `init.crs stop` command on all of the nodes.
2. Edit the `/var/opt/oracle/ocr.loc` file on all of the nodes and set the `ocrconfig_loc` parameter to `ocr_config_loc=new_location` where `new_location` is the new location of the OCR.
3. Restore the OCR from one of the automatic physical backups using the command `ocrconfig -restore`.
4. Run the `ocrcheck` command to verify the new OCR location.
5. Restart the CRS software on all of the nodes in your cluster by restarting each node.

## 4.3 Importing Oracle Cluster Registry Content

1. Shut down all the nodes in the cluster and restart one of them in single-user mode.
2. Import an OCR export file using the `ocrconfig -import` command from any node.
3. Start all the nodes in the cluster in multi-user mode.

## 5 Overview of SRVCTL for Administering RAC

The Server Control (SRVCTL) utility is installed on each node by default. You can use SRVCTL to start and stop the database and instances, manage configuration information, and to delete or move instances and services. SRVCTL also manages configuration information.

Some SRVCTL operations store configuration information in the Oracle Cluster Registry (OCR). SRVCTL performs other operations, such as starting and stopping instances, by sending requests to the Cluster Ready Services daemon (CRSD), which then starts or stops the Cluster Ready Services (CRS) resources.

### 5.1 Using Services in Real Application Clusters Environments

Services are groups or classifications of applications that comprise business components that extend and respond to application workloads. Examples of services are Accounts Payable (AP), Customer Relationship Management (CRM), etc. Services are a generic feature of the Oracle 10g Database; you can use services in both single-instance Oracle database and in Real Application Cluster (RAC) deployments.

You assign services to run on one or more instances, and alternate instances can serve as backup instances in case the primary instance fails. If a primary instance fails, then Oracle moves the service from the failed instance to a surviving alternate instance. Services enable you to model and deploy both planned and unplanned operations for all types of high availability or disaster recovery scenarios. During outages, RAC automatically restarts key components. Components that are eligible for automatic restart include instances, Oracle Net Services listeners, and the database as well as several database components.

### 5.2 Guidelines for Using SRVCTL in Real Application Clusters

Guidelines for using SRVCTL are as follows:

- To use SRVCTL to change your RAC database configuration, log in to the database as the oracle user. Members of the DBA group can start and stop the database.

- Only use the version of SRVCTL that is provided with Oracle Database 10g on RAC databases that are created or upgraded for Oracle Database 10g.
- Always use SRVCTL from the ORACLE\_HOME of the database that you are administering.
- SRVCTL does not support concurrent executions of commands on the same object. Hence only run one SRVCTL command at a time for each database, service, or any other objects.

### 5.3 Obtaining Command Line Help for SRVCTL

To see help for all SRVCTL commands, from the command line enter:

```
srvctl -h
```

To see the command syntax and a list of options for each SRVCTL command, from the command line enter:

```
srvctl command (or verb) object (or noun) -h
```

To see the SRVCTL version number enter: `srvctl -V`

### 5.4 SRVCTL Command Syntax and Options

SRVCTL commands, objects, and options are case sensitive. Database, instance, and service names are case insensitive and case preserving. SRVCTL interprets the following command syntax:

```
srvctl command object [options]
```

- `srvctl` is the command to start the SRVCTL utility.
- `command` is a verb such as `start`, `stop`, or `remove`.
- `object` is an object or target on which SRVCTL performs the command, such as `database` or `instance`. `options` extend the use of a preceding command combination to include additional parameters for the command. For example, the `-i` option indicates that a comma-delimited list of instance names follows; sometimes the `-i` option only permits one value. The `-n` option indicates that a node name or a comma-delimited list of node names follows. `-q` prompts for user credentials.

### 5.5 SRVCTL Cluster Database Configuration Tasks

The database configuration tasks are:

- Add, modify, and delete cluster database configuration information.
- Add an instance or a service to, and delete an instance or service from the configuration of a cluster database.
- Move instances and services in a cluster database configuration and modify service configurations.
- Set and unset the environment for an instance or service in a cluster database configuration.
- Set and unset the environment for an entire cluster database in a cluster database configuration.

Although you can cancel running SRVCTL commands by entering Control-C at the command line, you may **corrupt** your configuration data by doing this. You are strongly advised not to terminate SRVCTL in this manner.

### 5.6 SRVCTL General Cluster Database Administration Tasks

The general database administration tasks are:

- Start and stop cluster databases
- Start and stop cluster database instances
- Start, stop, and relocate cluster database services
- Obtain statuses of cluster databases, cluster database instances, or cluster database services

## 5.7 SRVCTL Node-Level Tasks

The node-level tasks are:

- Adding and deleting node level applications.
- Setting and unsetting the environment for node-level applications.
- Administering node applications and ASM instances.
- Starting and stopping a group of programs that includes virtual IP addresses, listeners, Oracle Notification Services, and Oracle Enterprise Manager agents (for maintenance purposes).

## 6 Summary:

Storage Foundation for Oracle RAC complements Oracle 10g Cluster Ready Services(CRS) and provides a numerous benefits to the customer with its built-in volume management, file system and clustering components. While it is important for a customer upgrading from a 9i RAC to a 10g RAC environment to understand CRS, it is definitely not a substitute to Storage Foundation for Oracle RAC.

## APPENDIX A - SRVCTL commands to control CRS resources

### SRVCTL ADD

The SRVCTL `add` command adds the configuration and the CRS applications to the OCR for the cluster database, named instances, named services, or for the named nodes. To execute `srvctl add` operations, you must be logged in as the database administrator and be the Oracle account owner on UNIX-based systems, or you must be logged on as a user with Administrator privileges on Windows-based systems.

When adding an instance, the name that you specify with `-i` must match the `ORACLE_SID` parameter. The database name given with `-d db_unique_name` must match the `DB_UNIQUE_NAME` initialization parameter setting. If `DB_UNIQUE_NAME` is unspecified, then match the `DB_NAME` initialization parameter setting. The default setting for `DB_UNIQUE_NAME` uses the setting for `DB_NAME`. Also, the domain name given with `-m db_domain` must match the `DB_DOMAIN` setting.

#### **srvctl add database**

Adds a database configuration to your cluster database configuration.

```
srvctl add database -d crm -o /ora/ora10
```

#### **srvctl add instance**

Adds a configuration for an instance to your cluster database configuration.

```
srvctl add instance -d crm -i crm01 -n gm01
srvctl add instance -d crm -i crm02 -n gm02
srvctl add instance -d crm -i crm03 -n gm03
```

#### **srvctl add service**

Adds services to a database and assigns them to instances. If you have multiple instances of a cluster database on the same node, then always use only one instance on that node for all of the services that node manages. Also, you can use the

#### **srvctl config database**

Displays the configuration for a RAC database or lists all configured databases.

```
srvctl config database
```

```
srvctl config service
```

Displays the configuration for a service.

```
srvctl config service -d crm -s crm
```

### SRVCTL ENABLE

The SRVCTL `enable` command enables the named object so that it can run under CRS for automatic startup, failover, or restart. The CRS application supporting the object may be up or down to use this function. Enable is the default value. If the object is already enabled, then the command is ignored. Enabled objects can be started, and disabled objects cannot be started

#### **srvctl enable database**

Enables CRS resources for a database and enables the database's instances if the database was previously disabled.

```
srvctl enable database -d crm
```

#### **srvctl enable instance**

Enables an instance for CRS. If all instances are disabled, then enabling an instance also enables the database.

```
srvctl enable instance -d crm -i "crm1,crm2"
```

#### **srvctl enable service**

Enables a service for CRS. Enabling an entire service also affects the enabling of the service over all the instances by enabling the service at each one. When the entire service is already enabled, a `srvctl enable service` operation does not affect all the instances and enable them. Instead, this operation returns an error. Therefore, you cannot always use the entire set of service operations to manipulate the service indicators for each instance.

```
srvctl enable service -d crm -s crm
```

## **SRVCTL DISABLE**

Disables a specified object (cluster database, database instance). When you issue the disable command, the object is disabled and unavailable to run under CRS for automatic startup, failover, or restart.

#### **srvctl disable database**

Disables a cluster database and its instances.

```
srvctl disable database -d mybd1
```

#### **srvctl disable instance**

Disables an instance. If the instance that you disable with this command is the last enabled instance, then this operation also disables the database.

```
srvctl disable instance -d crm -i "crm1,crm3"
```

#### **srvctl disable service**

Disables a service. Disabling an entire service affects all the instances, disabling each one. When the entire service is already disabled, a `srvctl disable service` operation on the entire service affects all the instances and disables them; it just returns an error. This means that you cannot always use the entire set of service operations to manipulate the service indicators for each instance.

The following example globally disables two services:

```
srvctl disable service -d crm -s crm,marketing
```

The following example disables a service running on the preferred named instance and results in running a service on one less instance:

```
srvctl disable service -d crm -s crm -i crm1
```

## **SRVCTL START**

Starts CRS enabled, non-running applications for the database, all or named instances, all or named service names, or node-level applications. For the `start` command, and for other operations that use a connect string, if you do not provide a connect string, then SRVCTL uses `"/ as sysdba"` to perform the operation. To run such operations, the owner of the `oracle` binary executables must be a member of the OSDBA group, and users running the commands must be in the OSDBA group also.

### **srvctl start database**

Starts a cluster database and its enabled instances. An example of this command is:

```
srvctl start database -d crm -o open
```

### **srvctl start instance**

Starts instances in the cluster database. An example of this command is:

```
srvctl start instance -d crm -i "crm1,crm4"
```

### **srvctl start service**

Starts a service or multiple services on the specified instance. The `srvctl start service` command will fail if you attempt to start a service on an instance if that service is already running on its maximum number of instances, that is, its number of preferred instances. You may move a service or change the status of a service on an instance with the `srvctl modify service` and `srvctl relocate service` commands.

```
srvctl start service -d crm -s crm
```

The following example starts a named service on a specified instance:

```
srvctl start service -d crm -s crm -i crm2
```

## **SRVCTL STOP**

Stops the CRS applications for the database, all or named instances, all or named service names, or node level applications. Only CRS applications that are starting or running are stopped. Objects running outside of CRS are not stopped. Stops node-level applications and all dependent CRS applications on the node.

### **srvctl stop database**

Stops a database, its instances, and its services. An example of this command is:

```
srvctl stop database -d crm
```

### **srvctl stop instance**

Stops instances and stops all enabled and non-running services that have these instances as either preferred or available instances. An example of this command is:

```
srvctl stop instance -d crm -i crm1
```

### **srvctl stop service**

Stops one or more services globally across the cluster database, or on the specified instance. An example of this command is:

```
srvctl stop asm -n crmnode1 -i asm1
```

The following example stops a service globally across a cluster database:

```
srvctl stop service -d crm -s crm
```

The following example stops a service on a specified instance:

```
srvctl stop service -d crm -s crm -i crm2
```

## **SRVCTL RELOCATE**

Relocates the named service names from one named instance to another named instance. The `srvctl relocate` command works on only one source instance and one target instance at a time, relocating a service

from a single source instance to a single target instance. The target instance must be on the preferred or available list for the service. The relocated service is temporary until you modify the configuration.

#### **srvctl relocate service**

Temporarily relocates a service member to run on another instance. To temporarily relocate a named service member from `crm1` to `crm3`:

```
srvctl relocate service -d crm -s crm -i crm1 -t crm3
```

### **SRVCTL STATUS**

Displays the current state of a named database, instances, services, or node applications.

#### **srvctl status database**

Obtains the status of instances and their services. An example of this command is:

```
srvctl status database -d crm -v
```

#### **srvctl status instance**

Obtains the status of instances. An example of this command is:

```
srvctl status instance -d crm -i "crm1,crm2" -v
```

#### **srvctl status service**

Obtains the status of a service. The following example obtains the status of a named service globally across the database:

```
srvctl status service -d crm -s crm -v
```

### **SRVCTL GETENV**

Gets and displays values for the environment from the configuration file. Use SRVCTL with the set, get, and unset environment configuration verbs to administer the environment configurations for databases, instances, services, and node applications.

#### **srvctl getenv database**

Displays the cluster database environment values. The following example gets the environment configuration for a cluster database:

```
srvctl getenv database -d crm
```

#### **srvctl getenv instance**

Gets the values for an instance environment configuration. The following example sets the environment configuration for an instance:

```
srvctl getenv instance -d -crm -i instance1
```

#### **srvctl getenv service**

Gets the values for a service environment configuration. Use the `srvctl getenv service` command with the following syntax:

```
srvctl getenv service -d db_unique_name -s service_name [-t name_list]
```

The following example lists all environment variables for a service:

```
srvctl getenv service -d crm -s crm
```

## SRVCTL REMOVE

Removes the configuration, the CRS applications for the node (including the virtual IP address, the Oracle Enterprise Manager agent, the GSD, and the listeners), the database, named instances, or the named services from the cluster database.

Environment settings for the object are also removed.

If you do not use the force flag (-f), then Oracle prompts you to confirm whether to proceed. If you use the force (-f) option, then the remove operation proceeds without prompting and continues processing even when it encounters errors. Even when the CRS resources cannot be removed, the OCR configuration is removed, so that the object now appears not to exist, but there are still CRS resources. Use the -f option with extreme caution because this could result in an inconsistent OCR.

To use the remove verb, you must first stop the node applications, database, instance, or service for which you are specifying `srvctl remove`. Oracle recommends that you perform a disable operation before using this command, but this is not required.

### srvctl remove database

Removes a database configuration. An example of this command is:

```
srvctl remove database -d crm
```

### srvctl remove instance

Removes the configurations for an instance. An example of this command is:

```
srvctl remove instance -d crm -i crm01
```

### srvctl remove service

Removes the configuration for a service. An example of this command is:

```
srvctl remove service -d crm -s sales
```

## APPENDIX B - main.cf utilizing single cssd resource

```

include "types.cf"
include "CFSTypes.cf"
include "CVMTypes.cf"
include "OracleTypes.cf"
include "PrivNIC.cf"
include "/etc/VRTSvcs/conf/config/VVRTypes.cf"

cluster oracle_10g (
    UserNames = { admin = gJKcJEjGKfKKiSKeJH }
    Administrators = { admin }
    CredRenewFrequency = 0
    UseFence = SCSI3
    HacliUserLevel = COMMANDROOT
    CounterInterval = 5
)

system thor150 (
)

system thor151 (
)

group cvm (
    SystemList = { thor150 = 0, thor151 = 1 }
    AutoFailOver = 0
    Parallel = 1
    AutoStartList = { thor150, thor151 }
)

Application cssd-resource (
    Critical = 0
    StartProgram = "/opt/VRTSvcs/rac/bin/cssd-online"
    StopProgram = "/opt/VRTSvcs/rac/bin/cssd-offline"
    CleanProgram = "/opt/VRTSvcs/rac/bin/cssd-clean"
    MonitorProgram = "/opt/VRTSvcs/rac/bin/cssd-monitor"
    OnlineRetryLimit = 20
)

CFSMount crs_ora_mnt (
    Critical = 0
    MountPoint = "/ora_crs"
    BlockDevice = "/dev/vx/dsk/crs_oradg_150_151/crs_vol"
)

CFSMount orabin_mnt (
    MountPoint = "/oracle"
    BlockDevice = "/dev/vx/dsk/orabin_150_151/orabin_vol"
)

```

```

CFSMount oradata_mnt (
    MountPoint = "/oradata"
    BlockDevice = "/dev/vx/dsk/oradata_150_151/oradata_vol"
)

CFSQlogckd qlogckd (
    Critical = 0
)

CFSfsckd vxfsckd (
)

CVMCluster cvm_clus (
    CVMClustName = oracle_10g
    CVMNodeId = { thor150 = 0, thor151 = 1 }
    CVMTransport = gab
    CVMTimeout = 200
)

CVMVolDg crs_voldg (
    CVMDiskGroup = crs_oradg_150_151
    CVMVolume = { crs_vol }
    CVMActivation = sw
)

CVMVolDg orabin_voldg (
    CVMDiskGroup = orabin_150_151
    CVMVolume = { orabin_vol }
    CVMActivation = sw
)

CVMVolDg oradata_voldg (
    CVMDiskGroup = oradata_150_151
    CVMVolume = { oradata_vol }
    CVMActivation = sw
)

CVMVxconfigd cvm_vxconfigd (
    Critical = 0
    CVMVxconfigdArgs = { syslog }
)

PrivNIC ora_priv (
    Device = { qfe0 = 0, qfel = 1 }
    Address @thor150 = "10.1.1.150"
    Address @thor151 = "10.1.1.151"
    NetMask = "255.255.255.0"
)

crs_ora_mnt requires crs_voldg
crs_ora_mnt requires vxfsckd

```

```

crs_voldg requires cvm_clus
cssd-resource requires crs_ora_mnt
cssd-resource requires ora_priv
cssd-resource requires orabin_mnt
cssd-resource requires oradata_mnt
cvm_clus requires cvm_vxconfigd
orabin_mnt requires orabin_voldg
orabin_mnt requires vxfsckd
orabin_voldg requires cvm_clus
oradata_mnt requires oradata_voldg
oradata_mnt requires vxfsckd
oradata_voldg requires cvm_clus
qlogckd requires cvm_clus
vxfsckd requires qlogckd

// resource dependency tree
//
//   group cvm
//   {
//   Application cssd-resource
//   {
//       CFSSMount crs_ora_mnt
//       {
//           CVMVoldg crs_voldg
//           {
//               CVMCluster cvm_clus
//               {
//                   CVMVxconfigd cvm_vxconfigd
//               }
//           }
//       }
//       CFSfsckd vxfsckd
//       {
//           CFSQlogckd qlogckd
//           {
//               CVMCluster cvm_clus
//               {
//                   CVMVxconfigd cvm_vxconfigd
//               }
//           }
//       }
//   }
//   CFSSMount orabin_mnt
//   {
//       CVMVoldg orabin_voldg
//       {
//           CVMCluster cvm_clus
//           {
//               CVMVxconfigd cvm_vxconfigd
//           }
//       }
//   }
//   }

```

```

//          CFSfsckd vxfscd
//          {
//          CFSQlogckd qlogckd
//          {
//          CVMCluster cvm_clus
//          {
//          CVMVxconfigd cvm_vxconfigd
//          }
//          }
//          }
//          }
//          CFSMount oradata_mnt
//          {
//          CVMVolDg oradata_voldg
//          {
//          CVMCluster cvm_clus
//          {
//          CVMVxconfigd cvm_vxconfigd
//          }
//          }
//          CFSfsckd vxfscd
//          {
//          CFSQlogckd qlogckd
//          {
//          CVMCluster cvm_clus
//          {
//          CVMVxconfigd cvm_vxconfigd
//          }
//          }
//          }
//          }
//          PrivNIC ora_priv
//          }
//          }

```

## APPENDIX C- main.cf utilizing two cssd resources

```

include "types.cf"
include "CFSTypes.cf"
include "CVMTypes.cf"
include "OracleTypes.cf"
include "PrivNIC.cf"

cluster t158 (
    UserNames = { admin = bopHoj0lpKppNxpJom, vcs = GLMkLIII,
                 oracle = chhPidGchDhnGg }
    AllowNativeCliUsers = 1
    Administrators = { admin, vcs }

```

```

    HacliUserLevel = COMMANDROOT
    CounterInterval = 5
)

system thor158 (
)

system thor159 (
)

system thor188 (
)

system thor189 (
)

group Oracle1 (
  SystemList = { thor158 = 0, thor159 = 1, thor188 = 2, thor189 = 3 }
  AutoFailOver = 0
  Parallel = 1
  AutoStartList = { thor158, thor159, thor188, thor189 }
)

Application cssdl (
  Critical = 0
  StartProgram = "/opt/VRTSvcs/rac/bin/cssd-online"
  StopProgram = "/opt/VRTSvcs/rac/bin/cssd-offline stopcrs"
  CleanProgram = "/opt/VRTSvcs/rac/bin/cssd-clean"
  MonitorProgram = "/opt/VRTSvcs/rac/bin/cssd-monitor"
  OnlineRetryLimit = 20
)

CFSMount oradata_mnt (
  Critical = 0
  MountPoint = "/oradata"
  BlockDevice = "/dev/vx/dsk/oradatadg/oradatavol"
)

requires group cvm online local firm
cssdl requires oradata_mnt

// resource dependency tree
//
//   group Oracle1
//   {
//   Application cssdl
//   {
//       CFSMount oradata_mnt
//   }
// }

```

```

group Oracle2 (
  SystemList = { thor158 = 0, thor159 = 1, thor188 = 2, thor189 = 3 }
  AutoFailOver = 0
  Parallel = 1
  AutoStartList = { thor158, thor159, thor188, thor189 }
)

Application cssd2 (
  Critical = 0
  StartProgram = "/opt/VRTSvcs/rac/bin/cssd-online"
  StopProgram = "/opt/VRTSvcs/rac/bin/cssd-offline stopcrs"
  CleanProgram = "/opt/VRTSvcs/rac/bin/cssd-clean"
  MonitorProgram = "/opt/VRTSvcs/rac/bin/cssd-monitor"
  OnlineRetryLimit = 20
)

CFSMount oradata_mnt1 (
  Critical = 0
  MountPoint = "/oradata1"
  BlockDevice = "/dev/vx/dsk/oradatadg/oradatavoll"
)

requires group cvm online local firm
cssd2 requires oradata_mnt1

// resource dependency tree
//
//   group Oracle2
//   {
//     Application cssd2
//     {
//       CFSMount oradata_mnt1
//     }
//   }

group cvm (
  SystemList = { thor158 = 0, thor159 = 1, thor188 = 2, thor189 = 3 }
  AutoFailOver = 0
  Parallel = 1
  AutoStartList = { thor158, thor159, thor188, thor189 }
)

CFSMount oraOCR_mnt (
  Critical = 0
  MountPoint = "/oraOCR"
  BlockDevice = "/dev/vx/dsk/oradatadg/oraOCRvol"
)

```

```

CFSMount oraVOTE_mnt (
    Critical = 0
    MountPoint = "/oraVOTE"
    BlockDevice = "/dev/vx/dsk/oradatadg/oravotvol"
)

CFSMount orabin_mnt (
    Critical = 0
    MountPoint = "/orabin"
    BlockDevice = "/dev/vx/dsk/orabindg/orabinvol"
)

CFSQlogckd qlogckd (
    Critical = 0
)

CFSfsckd vxfsckd (
)

CVMCluster cvm_clus (
    CVMClustName = t158
    CVMNodeId = { thor158 = 0, thor159 = 1, thor188 = 2,
                  thor189 = 3 }
    CVMTransport = gab
    CVMTimeout = 200
)

CVMVolDg oraOCR_voldg (
    CVMDiskGroup = oradatadg
    CVMVolume = { oraOCRvol }
    CVMActivation = sw
)

CVMVolDg oraVOTE_voldg (
    CVMDiskGroup = oradatadg
    CVMVolume = { oravotvol }
    CVMActivation = sw
)

CVMVolDg orabin_voldg (
    CVMDiskGroup = orabindg
    CVMVolume = { orabinvol }
    CVMActivation = sw
)

CVMVolDg oradata1_voldg (
    CVMDiskGroup = oradatadg
    CVMVolume = { oradatavoll }
    CVMActivation = sw
)

```

```

CVMVolDg oradata_voldg (
    CVMDiskGroup = oradatadg
    CVMVolume = { oradatavol }
    CVMActivation = sw
)

CVMVxconfigd cvm_vxconfigd (
    Critical = 0
    CVMVxconfigdArgs = { syslog }
)

PrivNIC ora_privnic (
    Critical = 0
    Device = { qfe1 = 1, qfe0 = 0 }
    DeviceTag @thor189 = { qf_e-1 = 1, qfe-0 = 0 }
    Address @thor158 = "192.11.12.58"
    Address @thor159 = "192.11.12.59"
    Address @thor188 = "192.11.12.88"
    Address @thor189 = "192.11.12.89"
    NetMask = "255.255.255.0"
)

```

```

cvm_clus requires cvm_vxconfigd
oraOCR_mnt requires oraOCR_voldg
oraOCR_mnt requires vxfsckd
oraOCR_voldg requires cvm_clus
oraVOTE_mnt requires oraVOTE_voldg
oraVOTE_mnt requires vxfsckd
oraVOTE_voldg requires cvm_clus
orabin_mnt requires orabin_voldg
orabin_mnt requires vxfsckd
orabin_voldg requires cvm_clus
oradata1_voldg requires cvm_clus
oradata_voldg requires cvm_clus
qlogckd requires cvm_clus
vxfsckd requires qlogckd

```

```

// resource dependency tree
//
//   group cvm
//   {
//     CFSSMount oraOCR_mnt
//     {
//       CVMVolDg oraOCR_voldg
//       {
//         CVMCluster cvm_clus
//         {
//           CVMVxconfigd cvm_vxconfigd
//         }
//       }
//     }
//   }

```

```

//      }
//      CFSfsckd vxfscd
//      {
//          CFSQlogckd qlogckd
//          {
//              CVMCluster cvm_clus
//              {
//                  CVMVxconfigd cvm_vxconfigd
//              }
//          }
//      }
//  }
// CFSMount oraVOTE_mnt
//  {
//      CVMVolDg oraVOTE_voldg
//      {
//          CVMCluster cvm_clus
//          {
//              CVMVxconfigd cvm_vxconfigd
//          }
//      }
//      CFSfsckd vxfscd
//      {
//          CFSQlogckd qlogckd
//          {
//              CVMCluster cvm_clus
//              {
//                  CVMVxconfigd cvm_vxconfigd
//              }
//          }
//      }
//  }
// PrivNIC ora_privnic
// CFSMount orabin_mnt
//  {
//      CVMVolDg orabin_voldg
//      {
//          CVMCluster cvm_clus
//          {
//              CVMVxconfigd cvm_vxconfigd
//          }
//      }
//      CFSfsckd vxfscd
//      {
//          CFSQlogckd qlogckd
//          {
//              CVMCluster cvm_clus
//              {
//                  CVMVxconfigd cvm_vxconfigd
//              }
//          }
//      }
//  }

```

```
//      }
//    }
//  CVMVolDg oradata1_voldg
//    {
//      CVMCluster cvm_clus
//        {
//          CVMVxconfigd cvm_vxconfigd
//        }
//    }
//  CVMVolDg oradata_voldg
//    {
//      CVMCluster cvm_clus
//        {
//          CVMVxconfigd cvm_vxconfigd
//        }
//    }
//  }
```

## **VERITAS Software Corporation**

Corporate Headquarters  
350 Ellis Street  
Mountain View, CA 94043  
650-527-8000 or 866-837-4827

For additional information about VERITAS Software, its products, or the location of an office near you, please call our corporate headquarters or visit our Web site at [www.veritas.com](http://www.veritas.com).