# Oracle RAC for Linux Manager Book

http://www.database8.com

# 基于 LINUX 的 Oracle 10G RAC 管理维护手记

前言: 本文主要记录了自己在学习和管理维护 RAC 过程中的一点心得和笔记，鉴于网上安装文档参考资料相对详尽，故尽量对此过程减少笔墨。RAC 是一个相对复杂的体系，其高效稳定运行与主机/存储/网络等密不可分，因此文中很多认知难免存在偏颇疏漏之处，姑且抛砖引玉。藉此，希望能得到更多指正并不吝分享宝贵经验。

如不加特殊说明，后续提及环境皆为 Redhat AS 4/Oracle 10gR2.

在阅读过程或实际中有什么问题，欢迎与我交流。

Msn:kevin.yuan@msn.com.

Blog:http://www.easyora.net



目录:

一 RAC 相关以及基础知识

1.CRS 简介

从 Oracle 10G 开始，oracle 引进一套完整的集群管理解决方案---Cluster-Ready Services，它包括集群连通性. 消息和锁. 负载管理等框架. 从而使得 RAC 可以脱离第三方集群件，当然，CRS 与第三方集群件可以共同使用.

(1).CRS 进程

CRS 主要由三部分组成，三部分都作为守护进程出现

<1>CRSD：资源可用性维护的主要引擎. 它用来执行高可用性恢复及管理操作，诸如维护 OCR 及管理应用资源，它保存着集群的信息状态和 OCR 的配置，此进程以 root 权限运行.

<2>EVMD：事件管理守护进程.此进程还负责启动 racgevt 进程以管理 FAN 服务器端调用，此进程以 root 权限运行

<3>OCSSD：集群同步服务进程.管理集群节点的成员资格，它以 fatal 方式启动，因此进程发生故障将导致集群重启，以防止数据坏死.同时，CSS 还维护集群内的基本锁功能，以及负责监控 voting disk 的脑裂故障。它以 Oracle 权限运行

此外，还有一个进程 OPRCD，他是集群中的进程监视程序，仅当平台上的 CRS 不使用厂商群件时候才出现，且无论运行了多少实例，每个节点只会存在一组后台进程.

来看一下这几个守护进程:

```
rac1-> cat /etc/inittab
  ..............................
 # Run xdm in runlevel 5
 x:5:respawn:/etc/X11/prefdm -nodaemon
 h1:35:respawn:/etc/init.d/init.evmd run >/dev/null 2>&1 </dev/null
 h2:35:respawn:/etc/init.d/init.cssd fatal >/dev/null 2>&1 </dev/null
 h3:35:respawn:/etc/init.d/init.crsd run >/dev/null 2>&1 </dev/null
```

(2).Virtual IP Address

Oracle 10G RAC 下，有 3 个重要的 IP.

① Public IP ② Private IP ③ Vitual IP

Public IP 为数据库所在主机的公共网络 IP，Private IP 被用来私有高速互联，而 Oracle 较前版本，增加了一个虚拟 IP，用来节点发生故障时候更快的故障转移，oracle 利用每个节点的 lisnter 侦听 VIP，一旦发生故障，VIP 将进行实际的故障切换，从而在其他的可用的节点上保持联机，从而降低客户应用程序意识到节点故障所需要的时间。

VIP 与 Public IP 必须在同一个网段内。

(3).OCR，Voting disk

OCR(oracle 集群注册表)和 Voting disk(表决磁盘)是 CRS 下的两个重要组件，它们必须放在共享磁盘

上，以保证每个节点都能对其访问。

OCR 包含了针对集群的一些配置信息，诸如:集群数据库中的节点列表.CRS 应用程序.资源文件以及事件管理器的授权信息。他负责对集群内的资源追踪，从而获知资源正在哪里运行，应该可以在哪里运行。

Voting disk 用来解决 split-brain 故障:如果节点丢失了与集群中其他节点的网络连接，这些冲突由表决磁盘中的信息来解决。

2.ASM 相关

ASM (Automated Storage Management) 是 Oracle 10G 引入的一种文件类型，他提供了直接的 I/O 读写，是 RAC 体系下一套不错的对数据文件存储规划的方案. ASM 可以自动管理磁盘组，并提供数据冗余和优化. 后面章节会就 ASM 的管理以及 ASM 下的 RAC 管理，单独讲解.

3.RAC 存储/网络需求

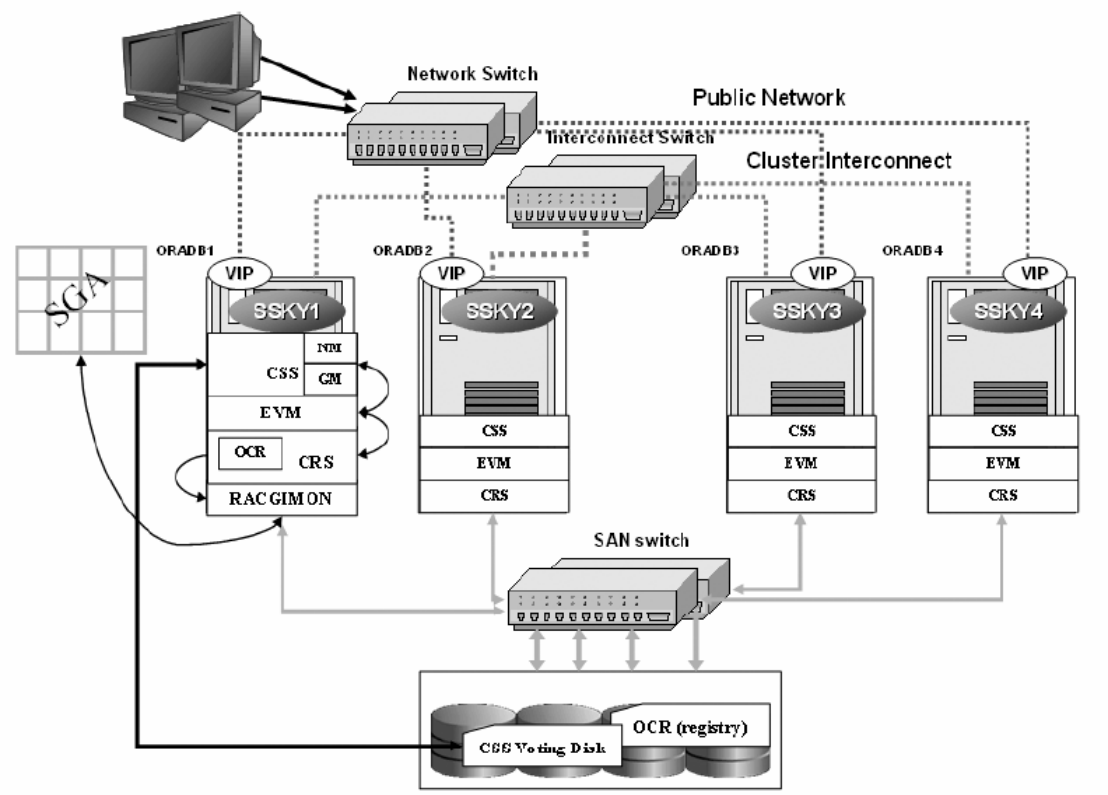


图 1.3 RAC 体系架构

(1).存储需求 

共享存储器是 RAC 的重要组件之一。它要求集群内的节点可以同时读写物理磁盘。目前，支持共享存储的文件类型也比较多，像 Oracle 自身提供的 ASM，OCFS2 以及三方提供的群集文件系统，都是可以选择的类型。

表 1.1.1 显示了 RAC 体系架构下各部分所支持的存储类型 (暂不考虑三方集群文件系统，就 ASM/raw device/OCFS2 和普通文件系统来说)

表 1.3.1　RAC 体系架构下各部分所支持的存储类型

| 类别 | 支持的存储类型 | 存储位置 | 备注 |
|---|---|---|---|
| Cluster 软件 | OCFS2，普通文件系统 | 共享磁盘/本地磁盘 | |
| OCR,Voting disk | OCFS2，raw device | 共享磁盘 | |
| 数据库软件 | OCFS2，普通文件系统 | 共享磁盘/本地磁盘 | |
| 数据库文件 | OCFS2，raw device，ASM | 共享磁盘 | |
| 归档日志文件 | OCFS2，ASM，普通文件系统 | 共享磁盘/本地磁盘 | |
| 备份/恢复文件 | OCFS2，ASM，普通文件系统 | 共享磁盘/本地磁盘 | |
| 闪回日志文件 | OCFS2，ASM | 共享磁盘 | |

(2).网络需求

每个节点主机上至少需要 **2** 张物理网卡，以便分配公有 **IP** 和私有 **IP** 地址。对于私有 **IP** 连接，每个集群节点通过专用高速网络连接到所有其他节点，目的在于集群上的节点和实例交换信息状态（锁信息，全局缓存信息等）。通过高速互联，**Cache Fusion** 得以实现。

在实际环境中，高速互联至少需要配置 **GB** 级的以太网，而且，最好不要使用交叉直连。较好的解决方案是节点间配置专用交换机，这样避免因为集群上一个节点宕掉而影响另外节点的正常工作。

4.其他

(1).后台进程

图 1.4.2 显示了 RAC 体系下的进程关系。

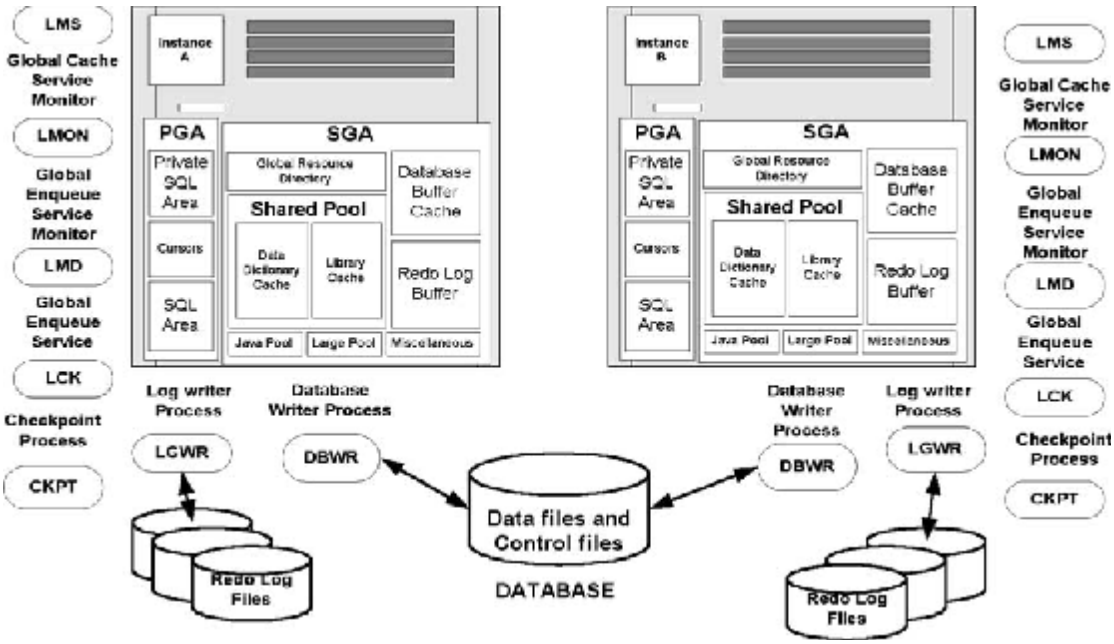


图 1.4.2 Backgroud Process in RAC 10g

由于要维护多个实例同时访问资源所必需的锁定，因此，同 single instance 相比，RAC 下增加了额外的一些进程。专门针对 RAC 的进程有如下几种：

1. **LMS(**Global Cache Service**)**　全局缓存服务进程

**LMS** 负责为缓存融合请求在实例间传递块。当一致性请求的时候，**LMS** 首先回滚块，创建块的读一致性映像（**CR**），然后将该一致性版本通过高速互联传递到处理此请求的远程实例中的前台进程

上，**LMS** 进程保证了在同一时刻只允许一个实例去更新数据块。

**LMS** 进程的数量由初始化参数 GCS_SERVER_PROCESSES 控制。Oracle 最大支持**36**个**LMS**进程（**0－9 and a－z**），该初始化参数默认值为2。

**2. LMD (Global Enqueue Service Daemon)** 全局队列服务守护进程

LMD 负责管理全局队列和全局资源访问，并更新相应队列的状态，此外还负责远程节点资源的请求与死锁的检测。**LMD** 与 **LMS** 进程互交工作，共同维护 **GRD**。

**3. LMON (Global Enqueue Service Monitor)** 全局队列服务监控器进程

**LMON** 是全局队列服务的监控器，他负责检查集群内实例的死亡情况并发起重新配置，当实例加入或者离开集群的时候，它负责重新配置锁和资源。

**4. LCK(Lock process)** 锁进程

**LCK** 管理那些不是缓存融合的请求，例如 **library cahe, row cache.**由于 **LMS** 进程提供了主要的锁管理功能，因此每个节点实例上只有一个 **LCK** 进程。

**5. DIAG (The Diagnostic Daemon**）诊断守护进程

DIAG 负责监控实例的健康状况并捕获进程失败的信息，并将失败信息写入用于失败分析，该进程自动启动且不需要人为调整，若失败则自动重新启动。

(2).缓存融合/缓存一致性

Cache Fusion 是 RAC 工作原理的一个中心环节.他的本质就是通过互联网络在集群内各节点的 SGA 之间进行块传递，从而避免了首先将块推送到磁盘，然后再重新读入其他实例的缓存中，从而最大限度的减少 I/O。当一个块被读入 RAC 环境中某个实例的缓存时，该块会被赋予一个锁资源（与行级锁不同），以确保其他实例知道该块正在被使用。之后，如果另一个实例请求该块的一个拷贝，而该块已经处于前一个实例的缓存内，那么该块会通过互联网络直接被传递到另一个实例的 SGA。如果内存中的块已经被改变，但改变尚未提交，那么将会传递一个 CR 副本。这就意味着，只要可能，数据块无需写回磁盘即可在各实例缓存之间移动，从而避免了同步多实例的缓存所花费的额外 I/O，由此，需要互联网络的速度是高速的，需要快于磁盘访问的速度.

GCS 负责维护全局缓冲区内的缓存一致性，LMS 进程是其主要组成部分。GES 确保同一时刻某个块上，只能有来自一个实例上的进程能对其修改,同时，并获得该块的当前版本和前映像，以及块所处的状态（NULL,,Shared, Exclusive），模式(local/gobal)。

GES 负责维护 dictionary cache 和 library cache 缓存一致性（这个与 LCK 是不同的）。由于存在某个节点上对数据字典的修改(比如 ddl 和 dcl 对 object 属性的修改),GES 负责同步各节点上的字典缓存,消除差异。GES 确保请求访问相同对象的多个实例间不会出现死锁。

GRD 包含了所有共享资源的当前状态信息，它由 GES 和 GCS 共同维护，GRD 贮存在内存中，被用来管理全局资源活动。比如：当一个实例第一次读取一个 block 到 SGA 的时候，该 block 的角色为 LOCAL，GCS 记录此状态到 GRD，一旦有另外的实例请求该块，GCS 会更新 GRD，将此块的角色由 LOCAL 变为 GLOBAL。

二 RAC 安装

不用把安装 RAC 看成是多么困难的一件事情.足够细心和耐性，充分的准备工作，然后加上一丁点运气，相信你能很快部署好一个 RAC 测试环境.当然，虚拟环境和实际环境的安装不尽相同，而且，生产系统环境的搭建需要经过缜密的规划和系统的测试.但大同小异，安装过程不该称为我们的绊脚石.

### 1.安装规划部署

安装之前需规划 RAC 系统各文件的存储类型。可以参考表 1.3.1。一个好的架构设计，会节省后期的管理维护成本。

### 2. 安装过程

安装过程可以参考 Oracle 联机文档中的 Install Guide （Vmware 下的安装可以参考 *Vincent Chan* 发表在 oracle 网站上的一文<使用 VMware Server 在 Oracle Enterprise Linux 上安装 Oracle RAC 10*g*>.） 文中讲的很详细，在此简单带过.简单列一下安装 RAC 的几个步骤

a) 配置系统内核参数以及环境
b) 配置共享存储/网络环境
c) 安装 CRS 软件
d) 安装 RDBMS 软件
e) 创建数据库以及配置其他

### 3.几点注意问题.

安装过程中，可能会碰到一些问题，诸如:

- /etc/hosts 文件没有合理配置，可能导致 ons 资源无法启动。
- 裸设备权限设置问题，导致采用其做存储类型的 crs 或者 asm 异常。
- 公共接口使用不可路由的 IP 地址,导致脚本$ORA_CRS_HOME/root.sh 执行报错.
- 节点间时间不同步，差异过大，可能导致安装集群件/数据库软件时候报错

……..

可能有很多人和我一样，使用的是虚拟机作为 RAC 学习的平台。特别提一下 vmware 下的时间同步问题，在我的环境下，两节点上时间差别很大.一开始采用 vmware-toolbox 工具同步宿主时间，效果不理想.后来在每个节点上放置一个小脚本，让他每隔一段时间以另一个节点为基准互相同步时间.这样，时间同步问题迎刃而解.在我的环境下，我设置每 20 秒同步一次时间.

```
rac1-> cat date.sh
#!/bin/sh
while true
do
rdate -s rac2>/dev/null 2>&1
sleep 20
done
```

三 RAC 管理维护

同 Single instance 相比，RAC 的管理维护要复杂一些。10G 给我们提供了一个强大的 EM 管理工具，将很多管理维护工作简单和界面化。我们也应当习惯使用 EM 来高效的完成更多的工作。本文以下部分，将暂不讨论 EM 方式的管理。

1.CRS 管理维护

(1).CRS 相关的接口命令

CRS 在 10G RAC 体系下有着举足轻重的作用。Oracle 也提供了一些命令接口让我们诊断维护它。

<1>CRS_*

10G RAC 下，有这么几组 crs_命令维护 CRS 资源。

```
[root@rac2 bin]# ls $ORA_CRS_HOME/bin|grep "crs_"|grep -v bin
crs_getperm
crs_profile
crs_register
crs_relocate
crs_setperm
crs_start
crs_stat
crs_stop
crs_unregister
```

下面分别讲述一下它们。

• 集群资源查询：CRS_STAT

可以用来查看 RAC 中各节点上 resources 的运行状况,Resources 的属性等。

例如使用－t 选项，检查资源状态：

```
[root@rac1 ~]# crs_stat –t
Name           Type           Target    State     Host
------------------------------------------------------------
ora.demo.db    application    ONLINE    ONLINE    rac2
ora....o1.inst application    ONLINE    ONLINE    rac1
ora....o2.inst application    ONLINE    ONLINE    rac2
ora....SM1.asm application    ONLINE    ONLINE    rac1
ora....C1.lsnr application    ONLINE    ONLINE    rac1
ora.rac1.gsd   application    ONLINE    ONLINE    rac1
ora.rac1.ons   application    ONLINE    ONLINE    rac1
ora.rac1.vip   application    ONLINE    ONLINE    rac1
ora....SM2.asm application    ONLINE    ONLINE    rac2
ora....C2.lsnr application    ONLINE    ONLINE    rac2
ora.rac2.gsd   application    ONLINE    ONLINE    rac2
ora.rac2.ons   application    ONLINE    ONLINE    rac2
ora.rac2.vip   application    ONLINE    ONLINE    rac2
```

利用-p 选项，获得资源配置属性。

```
[root@rac2 bin]# crs_stat -p ora.rac2.vip
```

```
NAME=ora.rac2.vip
TYPE=application
ACTION_SCRIPT=/opt/oracle/product/10.2.0/crs_1/bin/racgwrap
ACTIVE_PLACEMENT=1
AUTO_START=1
CHECK_INTERVAL=60
DESCRIPTION=CRS application for VIP on a node
.....
USR_ORA_STOP_MODE=immediate
USR_ORA_STOP_TIMEOUT=0
USR_ORA_VIP=192.168.18.112
```

利用-p 参数，获得资源权限。

```
[root@rac2 bin]# crs_stat -ls|grep vip
ora.rac1.vip   root          oinstall              rwxr-xr--
ora.rac2.vip   root          oinstall              rwxr-xr--
```

主要参数有-t/-v/-p/-ls/-f 等。具体可以参见 crs_stat –h

- 集群资源启动/停止 CRS_START/CRS_STOP

这组命令主要负责各个节点上 resources 的启动/停止。可以针对全局资源(例如：crs_stop –all，表示停止所有节点上的 resources),也可以针对节点上的某个特定的资源（例如：crs_start ora.rac2.ons,表示启动节点 rac2 上的 ONS）。

- 集群资源配置 CRS_REGISTER/CRS_UNREGISTER/CRS_PROFILE/CRS_SETPERM

这组命令主要负责集群资源的添加删除以及配置。

CRS_PROFILE:用来生成 resource 的 profile 文件(当然我们也可以手动编辑或者通过现有生成)，默认存放路径$ORA_CRS_HOME/crs/profile 目录下，加参数-dir 手动指定目录。默认名称为 resource_name.cap.

```
crs_profile -create resource_name -t application –a .. –r .. –o..
```

表 3.1 为 crs_profile 中参数配置说明（参数比较多，挑一些说吧）：

| 参数名称 | 说明 | 参数指令（以 create 为例） |
|---|---|---|
| NAME | 资源名称 | crs_profile –create resource_name |
| TYPE | 资源类型(application, generic) | crs_profile – create resource_name –t … |
| ACTION_SCRIPT | 用来管理 HA 方案脚本 | crs_profile – create resource_name –a … |
| ACTIVE_PLACEMENT | 资源贮存的位置/节点 | crs_profile –create resource_name –o –ap … |
| AUTO_START | 资源自启动 | crs_profile –create resource_name –o –as … |
| CHECK_INTERVAL | 资源监控间隔 | crs_profile –create resource_name –o –ci … |
| FAILOVER_DELAY | 资源 failover 的等待时间 | crs_profile –create resource_name –o –fd … |
| FAILURE_INTERVAL | 资源重启尝试间隔 | crs_profile –create resource_name –o –fi … |

| | | |
|---|---|---|
| FAILURE_THRESHOLD | 资源重启尝试次数(最大 20 次) | crs_profile –create resource_name –o –ft … |
| HOSTING_MEMBERS | 资源启动或者 failover 的首要节点选择 | crs_profile –create resource_name –h … |
| PLACEMENT | 资源启动或者 failover 的节点选择模式( balanced、favored、restricted ) | crs_profile – create resource_name –p |
| REQUIRED_RESOURCES | 当前资源所依赖的资源 | crs_profile – create resource_name –r |
| RESTART_ATTEMPTS | 资源重配置之前的尝试启动次数 | crs_profile –create resource_name –o –ra … |
| SCRIPT_TIMEOUT | 等待 ACTION_SCRIPT 的结果返回时间 | crs_profile –create resource_name –o –st … |
| USR_ORA_VIP | Vip 地址 | crs_profile –create vip_name -t application –a $ORA_CRS_HOME/bin/uservip –o oi=…, ov=…, on=… |

*crs_profile –update resource_name …* 用来更新现有 profile (更新的只是 profile，而并不是对已经注册到 OCR 里面的资源属性的更改)

crs_register 负责将 resource 的注册到 OCR。注册的方法是先生成 profile，然后运行 crs_register resource [-dir …]命令，同时，crs_register 也具有 update resource 功能，具体办法可以更新 resource 对应的 profile 文件，然后运行 crs_register -u resource_name [-dir …] 或者直接发布 crs_register –update resource_name …

比如，我将 rac 节点上的 vip 改为手动启动。

```
[root@rac1 crs]# crs_register -update ora.rac1.vip -o as=0
[root@rac1 crs]# crs_stat -p ora.rac1.vip|grep AUTO_START
AUTO_START=0
```

crs_unregister 负责将 resource 从 ocr 中移除。必要时候需要加－f 参数。

crs_setperm 用来设置 resource 的权限（诸如设置 owner，用户的读写权限等），更改 owner 用-o 参数，更改 group 用-g，更改用户权限用-u，在此不多举例了。

<2>.CRSCTL

用 crsctl check crs，检查 crs 的健康情况。

```
[root@rac1 ~]# crsctl check crs
CSS appears healthy
CRS appears healthy
EVM appears healthy
```

用 crsctl 控制 CRS 服务

```
crsctl start|stop|enable|disable crs
```

用 crsctl 启动/停止 resource

```
[root@rac1 ~]# crsctl stop resources
Stopping resources.
Successfully stopped CRS resources
[root@rac1 ~]# crsctl start resources
```

```
Starting resources.
Successfully started CRS resources
```

用 crsctl 检查以及添加、删除 voting disk

下面讲述。

更多参见 crsctl help。

<3>SRVCTL

SRVCTL 是一个强大的 CRS 和 RDBMS 的管理配置工具。相关用法参照 srvctl -h

① srvctl add/delete .. 添加删除资源。譬如我们在进行数据库单实例迁移到 rac 的时候，可以用这个工具手工注册 database 或者 asm 实例到 OCR。

② srvctl status … 资源的状态监测

③ srvctl start/stop … 资源的启动/停止，这个可以和 crs_start/crs_stop 互交使用。

④ srvctl modify .. 重新定义资源的属性。

……..

(2).OCR 的管理维护

<1> OCR 的状态验证：

可以使用 ocrcheck 工具来验证 OCR 的状态以及空间使用情况。在 Lunix 下，/etc/oracle/ocr.loc 文件记录了 OCR 使用的设备情况。

```
[root@rac1]# ocrcheck
Status of Oracle Cluster Registry is as follows :
     Version                  :          2
     Total space (kbytes)     :     497896
     Used space (kbytes)      :       3996
     Available space (kbytes) :     493900
     ID                       :  958197763
     Device/File Name         : /dev/raw/raw5
                                Device/File integrity check succeeded
                                Device/File not configured
     Cluster registry integrity check succeeded
```

<2> 在线添加/删除 ocrmirror

OCR 支持一个镜像，添加/删除镜像可以在线完成，在任意一个 online 的节点上执行命令即可。

```
[root@rac1]#ocrconfig -replace ocrmirror /dev/raw/raw5
[root@rac1 oracle]# cat /etc/oracle/ocr.loc
#Device/file  getting replaced by device /dev/raw/raw5
ocrconfig_loc=/dev/raw/raw1
ocrmirrorconfig_loc=/dev/raw/raw5
```

可见，ocr.loc 被自动更新。

移除 ocr 或者镜像的时候，只要不带路径，即可。

当一个 crs 中存在 ocr 和镜像的时候，如果移除 ocr，镜像会自动转变成 ocr 的角色。

```
[root@rac1]# ocrconfig -replace ocr
[root@rac1]# cat /etc/oracle/ocr.loc
#Device/file /dev/raw/raw1 being deleted
ocrconfig_loc=/dev/raw/raw5
```

可以看到现在的 ocrconfig_loc 自动变为先前的 ocrmirrorconfig_loc 设备。

<3> 逻辑备份/恢复

备份命令：

```
ocrconfig –export  [ path ]
```

还原命令

```
ocrconfig –import  [ path ]
```

还原 OCR 的时候，需要停掉各节点 crs 服务。还原完成后，重新启动 CRS。(如果有必要，注意在每个节点分别修改 ocr.loc 的对应使用设备)

<4> 物理备份/恢复

CRSD 负责每 4 个小时进行一次 OCR 的备份，默认备份路径在$ORA_CRS_HOME/cdate/crs 下，可以使用 ocrConfig –showbackup 查看备份情况，如果想更改物理备份路径，可以使用 ocrconfig –backuploc [ path ] 来完成

物理恢复命令：

```
ocrconfig –restore [ path ]
```

同样,还原 OCR 的时候，需要停掉各节点 crs 服务。还原完成后，重新启动 CRS。(如果有必要，注意在每个节点分别修改 ocr.loc 的对应使用设备)

<5> ocrdump

ocrdump 可以将 ocr 信息导出成 ascii 文本，用于给 Oracle Supoort 提供检修。

命令如下：

```
ocrdump <filename>
```

(3).Voting disk 管理维护

Voting disk 的维护相对简单些。

<1> Votingdisk 状态查询

```
[root@rac1]# crsctl query css votedisk
 0.     0    /dev/raw/raw2
located 1 votedisk(s).
```

<2>在线添加、删除 votingdisk

Oracle 建议配置奇数个 votingdisk，添加/删除可以在线完成，在任意一个 online 的节点上执行命令即可。

添加 votingdisk 命令：

```
crsctl add    css votedisk <path> -force
```

删除 votingdisk 命令：

```
crsctl add    css votedisk <path> -force
```

<3>votingdisk 备份恢复

备份、恢复采用 dd 命令。恢复的时候，注意停掉各节点上的 CRS 服务。

2.RDBMS 管理维护

(1).spfile 以及相关参数说明

最普遍情况，节点共用同一个 spfile 文件，放置在共享存储上，而每个节点上，相应目录下有一个 pfile 文件，而这个 pfile 文件指向共享存储上的 spfile。

当我们需要修改某一节点上的 paremeter 的时候，需要显示的指定 sid，例如：

```
SQL>alter system set sga_target=1024M scope=spfile sid='rac1';
```

```
System Altered.
```

这样，节点 rac1 上的 sga_target 参数被修改，不会影响其余节点上的参数设置。如果不加 sid，默认为 sid＝'*'，也就是对所有节点生效。

RAC 下，有一些不同与单实例的参数，列举如下：

① cluster_database

一般情况下，该参数在 rac 各实例下应该设置为 true。在一些特别情况下，比如 upgrade 等，需要将该参数设置成 false。

② db_name/db_unique_name/instance_name

各节点 db_name 需要一致，db_unique_name 也需要一致(这与 standby 是不同的)。而 instance_name 配置成各个节点的实例名称。

③ instance_number

该参数表示节点上实例的实例号。

④ thread

该参数用来标示实例使用的 redo 线程。线程号与节点号/实例号没有直接关联。

⑤ local_listener

该参数用来手工注册监听。为解决 **ORA-12514** 错误，可以设置该参数。

⑥ remote_listener

该参数用来进行服务器端负载均衡配置。

⑦ cluster_interconnects

该参数用来指定集群中 IPC 通信的网络。如果集群中有多种网络用于高速互联，需要配置该参数。对于多个 IP 地址，用冒号将其隔开。对于集群中当前使用的互联地址，可以查询视图 gv$cluster_interconnects 或着 oradebug ipc 来查看。

⑧ max_commit_propagation_delay

该参数用于配置 SCN 的产生机制。在 rac 下,SCN 的同步有 2 种模式:(1) Lamport Scheme.该模式下，由 GES 管理 SCN 的传播同步，max_commit_propagation_delay 表示 SCN 同步所允许的最大时间。在该模式下，全局 SCN 并非完全同步，这在高并发的 OLTP 系统中，可能会对应用造成一定的影响。(2) Broadcast on Commit scheme. 该模式下，一旦任何一个实例上事务发布 commit，都立即同步 SCN 到全局。

在 10g R1 下，该参数默认数值为 700，即采用 Lamport Scheme 模式。而在 10g R2 下，该参数默认数值为 0，采用 Broadcast on Commit scheme 模式 (设置小于 700 的某一值，都将采用该模式) 。采用何种方式，可以从 alert.log 中获知。该参数值需要每个节点保持一致。

(2). Redo/Undo 管理

• RAC 下的 Redo 管理

同单实例的系统一样，每个节点实例都需要至少 2 组 logfile。各节点实例有自己独立的重做日志线程(由初始化参数 thread 定义)，例如：

```
SQL> select b.THREAD#,a.GROUP#,a.STATUS,a.MEMBER,b.BYTES,b.ARCHIVED,b.STATUS
from v$logfile a,v$log b where a.GROUP#=b.GROUP#;
THREAD# GROUP# STATUS  MEMBER                                      BYTES  ARCHIVED  STATUS
------------------ ------- ------------------------------------------------
1  1   STALE +DATA/demo/onlinelog/group_1.257.660614753  52428800 YES  INACTIVE
1  2         +DATA/demo/onlinelog/group_2.258.660614755  52428800  NO   CURRENT
2  3         +DATA/demo/onlinelog/group_3.265.660615545  52428800  NO   CURRENT
2  4  STALE  +DATA/demo/onlinelog/group_4.266.660615543  52428800 YES  INACTIVE
```

重做日志需要部署到共享存储中，必须保证可被所有的集群内的节点实例访问。当某个节点实例进行实例/介质恢复的时候，该节点上的实例将可以应用集群下所有节点实例上的重做日志文件(如果需要)，从而保证恢复可以在任意可用节点进行。

• RAC 下 alter system switch logfile 与 alter system archive log current 区别

alter system switch logfile 仅对当前发布节点上的对应 redo thread 进行日志切换并归档。

alter system archive log current 对集群内所有节点实例上的 redo thread 进行切换并归档（在节点实例可用情况下，分别归档到各节点主机的归档目的地，当节点不可用时候，该线程日志归档到命令发布节点的归档目的地）

• RAC 下的 Undo 管理

RAC 下的每个节点实例，也需要有自己单独的撤销表空间。由初始化参数 *.Undo_tablespace 指定。同 REDO 一样，UNDO 表空间也需要部署到共享存储，虽然每个节点上 UNDO 的使用是独立的，但需要保证集群内其他节点实例对其访问，以完成构造读一致性等要求。

```
SQL>alter system set undo_tablespace=undo1 sid='demo1';
SQL>alter system set undo_tablespace=undo2 sid='demo2';
```

(3).Archivelog/flashback 配置管理

在 RAC 下，Archivelog 可以放置到本地磁盘，也可以放置到共享存储。需要对 Archivelog 的放置有合理的部署，如果放置到本地磁盘，会增加备份恢复的复杂程度。

闪回区必须部署到共享存储上，开启前，需要配置 db_recovery_file_dest、db_recovery_file_dest_size、db_flashback_retention_target 等参数。

下面在一个非归档非闪回的 database 上，开始归档与闪回。

• 更改相关参数

```
SQL>alter system set log_archive_dest_1='location=/archive/demo1' sid='demo1';
System altered
SQL> alter system set log_archive_dest_1='location=/archive/demo2' sid='demo2';
System altered
SQL> alter system set db_recovery_file_dest_size=512M;
System altered
SQL> alter system set db_recovery_file_dest='+DG1';
System altered
```

• 停掉所有节点实例.开启过程在一个实例上完成。

```
rac1-> srvctl stop instance -d demo -i demo1
rac1-> srvctl stop instance -d demo -i demo2
rac1-> sqlplus /nolog
SQL*Plus: Release 10.2.0.1.0 - Production on Sun Aug 3 22:06:50 2008
Copyright (c) 1982, 2005, Oracle.  All rights reserved.
SQL> conn /as sysdba
Connected to an idle instance.
SQL> startup mount;
ORACLE instance started.
Total System Global Area  167772160 bytes
Fixed Size                  1218316  bytes
Variable Size             100665588 bytes
Database Buffers           62914560 bytes
```

```
Redo Buffers              2973696 bytes
Database mounted.
SQL> alter database archivelog;
Database altered.
SQL> alter database flashback on;
Database altered.
SQL> alter database open;
Database altered.
SQL> select NAME,LOG_MODE,FLASHBACK_ON from v$database;
NAME      LOG_MODE     FLASHBACK_ON
--------- ------------ ------------------
DEMO      ARCHIVELOG   YES
```

10G 下，开启归档和闪回并不需要像 9i 那样，设置初始化参数 cluster_database＝false.这无疑简化了操作。

(4).ASM 下的 RAC 管理

• ASM 下的参数文件

RAC 下，每个节点上有运行有一个 ASM 实例，而 rdbms instance 就运行在这个 asm 实例上。Asm 实例是本地的。同 rdbms 实例一样，他需要有参数文件，参数文件在每个节点的相应目录下。

下面是我的 ASM 实例下的 pfile 文件：

```
cluster_database=true
background_dump_dest=/opt/oracle/admin/+ASM/bdump
core_dump_dest=/opt/oracle/admin/+ASM/cdump
user_dump_dest=/opt/oracle/admin/+ASM/udump
instance_type=asm
large_pool_size=12M
remote_login_passwordfile=exclusive
asm_diskgroups='DG1'
+ASM2.instance_number=2
+ASM1.instance_number=1
```

简单介绍几个 asm 实例中比较重要的参数:

instance_type：用来说明实例是ASM 还是RDBMS 类型

asm_diskgroups：ASM磁盘组，asm实例启动的时候会自动mount

asm_diskstring：该参数用来说明能够创建diskgroup的磁盘设备,默认值是NULL

asm_power_limit:该参数用来设置进程 ARBx 的数量，负责控制负载平衡操作的速度。取值 从 0 到 11。默认值为1。

• 用于记录ASM实例信息的数据字典。

V$ASM_DISK/ V$ASM_DISK_STAT：记录可以被ASM实例识别的磁盘信息，但这些磁盘并不一定是正在被实例使用的。

V$ASM_DISKGROUP/ V$ASM_DISKGROUP_STAT：记录asm下的diskgroup信息。

V$ASM_ALIAS：记录diskgroup文件的别名信息。

V$ASM_FILE：记录diskgroup中的文件信息。

V$ASM_OPERATION：记录ASM实例中当前运行的一个长时间操作信息。

V$ASM_TEMPLATE：记录diskgroup模板。

V$ASM_CLIENT：记录使用该asm实例下的diskgroup的rdbms实例信息。

• RAC下ASM磁盘组/文件管理操作

<1>.RAC下在线添加、删除磁盘组

在一个节点上添加diskgroup，集群上另外的节点并不会自动mount新添加的diskgroup，需要手动执行。

节点1：

```
SQL> show parameter asm_diskgroups
NAME                                 TYPE        VALUE
------------------------------------ -----------
asm_diskgroups                        string      DATA, DG1
SQL>CREATE DISKGROUP DATA2  NORMAL REDUNDANCY
    FAILGROUP DATA2_gp1 DISK '/dev/raw/raw6'
    FAILGROUP DATA2_gp2 DISK '/dev/raw/raw7';
Diskgroup created.
SQL> show parameter asm_diskgroups
NAME                                 TYPE        VALUE
------------------------------------ -----------
asm_diskgroups                        string      DATA, DG1, DATA2
```

此时观察节点 2，新加的磁盘组没有被 mount。

```
SQL> show parameter asm_diskgroups
NAME                                 TYPE        VALUE
------------------------------------------------
asm_diskgroups                         string       DATA, DG1
SQL>select group_number,type,state,type,total_mb,free_mb from
v$asm_diskgroup_stat;
GROUP_NUMBER   STATE       TYPE     TOTAL_MB    FREE_MB
-------------- -------------- ------------------------
         1     CONNECTED   EXTERN       5726       4217
         2     CONNECTED   EXTERN        415        297
         0    DISMOUNTED                    0          0
SQL>alter diskgroup DATA2 mount;
```

删除 diskgroup 时，保留一个节点 diskgroup 为 mount 状态，将其余节点上的 diskgroup dismount，然后执行删除命令。

<2>.在线添加、删除磁盘

RAC 下在线添加磁盘与删除磁盘与单实例并不差别。需要注意该操作会引起磁盘组的重新平衡，并确保删除磁盘的时候该磁盘组有足够的剩余空间。

节点 1：

```
SQL> alter diskgroup dg6 add disk '/dev/raw/raw7' name dg6_disk7;
Diskgroup altered.
```

节点 2 上查询：

```
SQL>select GROUP_NUMBER,path,NAME,MOUNT_STATUS,HEADER_STATUS,MODE_STATUS,
STATE from v$asm_disk_stat where NAME is not null;
 GROUP_NUMBER PATH     NAME       MOUNT_S HEADER_STATU MODE_ST STATE
```

```
------------ ---------------- ---------- ------- ------------ -------
           1 /dev/raw/raw3    DATA_0000  CACHED  MEMBER       ONLINE  NORMAL
           2 /dev/raw/raw4    DG1_0000   CACHED  MEMBER       ONLINE  NORMAL
           3 /dev/raw/raw6    DG6_0001   CACHED  MEMBER       ONLINE  NORMAL
           3 /dev/raw/raw7    DG6_DISK7  CACHED  MEMBER       ONLINE  NORMAL
```

删除磁盘在某一节点操作即可，不做举例验证。

关于 ASM 的更多管理命令，就不多列举了。

### 3.Database 备份/恢复

RAC 下的备份恢复跟单实例的备份恢复实质上没有太大的差别，需要注意的是备份/恢复的时候当前节点对所有数据文件/归档日志的可见。在一个数据文件和归档日志全部放在共享存储上的 RAC 系统，备份与恢复过程与单实例下的几乎一样。而归档日志如果采用的是本地磁盘，就需要另加注意。下面分别来模拟这个备份恢复过程。

(1).Archivelog 对各节点可见的备份/恢复

在这种模式下，备份恢复可以在任意一个可用节点执行即可，跟单实例并不太大区别。

• 对 database 进行备份

```
RMAN>run
{allocate channel oradem0 type disk;
backup database format '/backup/database/db_%s_%p_%t' plus archivelog format
'/backup/database/arch_%s_%p_%t' delete input;
backup current controlfile format '/backup/database/contr_%s_%p_%t';}
```

```
allocated channel: oradem0
channel oradem0: sid=130 instance=demo2 devtype=DISK
Starting backup at 03-MAY-08
current log archived
channel oradem0: starting archive log backupset
channel oradem0: specifying archive log(s) in backup set
input archive log thread=1 sequence=5 recid=70 stamp=661823848
input archive log thread=1 sequence=6 recid=72 stamp=661823865
……..
Finished backup at 03-MAY-08
released channel: oradem0
```

• 添加数据，用于测试恢复效果

```
SQL> create table kevinyuan.test_b as select * from dba_data_files;
Table created
SQL> alter system switch logfile;
System altered
SQL> insert into kevinyuan.test_b select * from dba_data_files;
6 rows inserted
SQL> commit;
Commit complete
```

```
SQL> select count(*) from kevinyuan.test_b;
  COUNT(*)
----------
       12
```

• 模拟故障/恢复

```
RMAN> run
{restore controlfile from '/backup/database/contr_16_1_661823935';
 sql 'alter database mount';
 restore database;
 recover database;
 sql 'alter database open resetlogs'; }
```

Starting restore at 04-MAY-08

allocated channel: ORA_DISK_1

……..

archive log filename=+DATA/demo/onlinelog/group_4.266.660615543 thread=2 sequence=11

archive log filename=+DATA/demo/onlinelog/group_3.265.660615545 thread=2 sequence=12

media recovery complete, elapsed time: 00:00:00

Finished recover at 04-MAY-08

sql statement: alter database open resetlogs

• 恢复完毕，来看一下验证数据：

```
SQL> select count(*) from kevinyuan.test_b;
  COUNT(*)
----------
       12
```

(2). Archivelog 对各节点不可见的备份/恢复

如果 arhivelog 采用本地磁盘,归档日志并不是对任意节点可见。备份 archivelog 的时候，如果采用和上述类似的备份方案，必然会导致一些归档日志由于无法 access 而抛出异常。可以采取如下的备份方式，目的就是使得备份通道能够 access 所有的数据字典中记录的归档日志信息。

恢复的时候， copy 所有节点产生的相关备份片/集和所需的归档日志到待恢复节点，在一个节点上执行 restore/recover 操作即可。

模拟一下这个操作。

```
SQL> alter system set log_archive_dest_1='location=/archive/demo1/'
sid='demo1';
System altered
SQL> alter system set log_archive_dest_1='location=/archive/demo2/'
sid='demo2';
System altered
```

• 备份数据库

```
RMAN>run
{allocate channel orademo1 type disk connect sys/kevinyuan@demo1;
 allocate channel orademo2 type disk connect sys/kevinyuan@demo2;
```

```
backup database format '/backup/database/db_%s_%p_%t' plus archivelog format
'/backup/database/arch_%s_%p_%t' delete input;
backup current controlfile format '/backup/database/contr_%s_%p_%t;}
```

allocated channel: orademo1
channel orademo1: sid=133 instance=demo1 devtype=DISK
allocated channel: orademo2
channel orademo2: sid=151 instance=demo2 devtype=DISK
Starting backup at 04-MAY-08
current log archived
channel orademo2: starting archive log backupset
channel orademo2: specifying archive log(s) in backup set
input archive log thread=2 sequence=4 recid=89 stamp=661826286
…….
channel orademo1: finished piece 1 at 04-MAY-08
piece handle=/backup/database/contr_28_1_661826504 tag=TAG20080504T004130 comment=NONE
channel orademo1: backup set complete, elapsed time: 00:00:09
Finished backup at 04-MAY-08
released channel: orademo1
released channel: orademo2

• COPY 节点 2 上的备份文件和归档日志到节点 1 相应目录下。

```
rac2-> scp /backup/database/* rac1:/backup/database/
rac2-> scp /archive/demo2/*   rac1:/archive/demo2/
```

• 恢复 database

```
RMAN>run
{restore database;
 recover database;
 sql 'alter database open';}
```

starting restore at 04-MAY-08
using target database control file instead of recovery catalog
allocated channel: ORA_DISK_1
channel ORA_DISK_1: sid=147 instance=demo1 devtype=DISK
channel ORA_DISK_1: restoring control file
channel ORA_DISK_1: restore complete, elapsed time: 00:00:20
……
archive log filename=+DATA/demo/onlinelog/group_3.265.660615545 thread=2 sequence=7
archive log filename=+DATA/demo/onlinelog/group_4.266.660615543 thread=2 sequence=8
media recovery complete, elapsed time: 00:00:06
Finished recover at 04-MAY-08
sql statement: alter database open

至此，恢复完成。

生产库的备份需要缜密部署与模拟测试，不同的数据库类型也需要制定不同的方案实现。对 DATABASE 来说，备份重于泰山，不能抱有任何侥幸心理。

四.故障切换/负载平衡配置

1.Service

服务是 rac 体系中相当重要的概念，它为应用提供高可用和多样化的解决方案。实际中，我们可以创建不同性质的 servie 来满足我们应用的不同需求。

10gR2 下，可以通过以下几个方式创建服务。

(1).使用 dbca

(2).使用 srvctl

```
node1->srvctl add service -d demo -s srv_1 -r node1 -a node2
node1-> srvctl start service -d demo -s srv_1
node1-> crs_stat -t
Name          Type          Target    State     Host
------------------------------------------------------------
ora.demo.db    application    ONLINE    ONLINE    node1
ora....o1.inst application    ONLINE    ONLINE    node1
ora....o2.inst application    ONLINE    OFFLINE
ora....rv_1.cs application    ONLINE    ONLINE    node1
ora....mo1.srv application    ONLINE    ONLINE    node1
 ------------------------------------
SQL> show parameter service
NAME                                 TYPE        VALUE
------------------------------------ ----------- -----------
service_names                         string      demo,srv_1
```

(3).使用 dbms_service 命令创建

10g 提供了 dbms_service 用于管理服务并进行功能扩展.

```
SQL>EXEC    DBMS_SERVICE.CREATE_SERVICE(SERVICE_NAME=>'srv_2',NETWORK_NAME=>'
srv_2');
PL/SQL procedure successfully completed

SQL> exec DBMS_SERVICE.START_SERVICE(service_name => 'srv_2',instance_name =>
'demo1');
PL/SQL procedure successfully completed

SQL> show parameter service
NAME                                 TYPE    VALUE
------------------------------------ ----------- -----------
service_names                         string      demo,srv_2
```

(4).其他等..

不管采用哪种方式，实质上都是通过添加 service_names 而向 lisnter 动态注册服务.

2. failover

RAC 为应用提供了高性能和高可用的服务，对用户来讲，核心的功能便是 failover 与 load banance.

在 10gR2 版本里，Failover 的实现方式有两种，一种是 TAF(Transparent Application Failover), 一种是

FCF(Fast Connection Failover).

(1).TAF 以及实现

TAF 是 net 层透明故障转移,是一种被动的故障转移方式, 依赖于 VIP.可以通过客户端和服务器端配置 TAF 的策略.

① client 端 TAF 配置

以下是一个简单的具有 TAF 功能的 tnsnames.ora 内容

```
demo =
  (DESCRIPTION =
    (FAILOVER=ON)
    (ADDRESS=(PROTOCOL=TCP)(HOST=10.194.129.145)(PORT=1521))
    (ADDRESS=(PROTOCOL=TCP)(HOST=10.194.129.146)(PORT=1521))
    (CONNECT_DATA =
       (SERVICE_NAME = demo)
       (SERVER=DEDICATED)
       (FAILOVER_MODE=(TYPE=SELECT)
                      (METHOD=BASIC)
                      (RETRIES=50)
                      (DELAY=5)
     )
    )
  )
```

控制 TAF 的参数说明：

| 参数 | 描述 |
|---|---|
| FAILOVER | Failover 控制开关（on/off），如果为 off，不提供故障切换功能，但连接时会对 address 列表进行依次尝试，直到找到可用为止 |
| TYPE | 两种类型：session /select<br>Session: 提供 session 级别的故障切换。<br>Select：提供 select 级别的故障切换，切换过程对查询语句透明，但事物类处理需要回滚操作 |
| METHOD | 两种类型:basic/preconnect<br>Basic:client 同时只连接一个节点，故障切换时跳转到另外节点<br>Preconnect：需要与 backup 同时使用，client 同时连接到主节点和 backup 节点 |
| BACKUP | 采用 Preconnect 模式的备用连接配置 |
| RETRIES | 故障切换时重试次数 |
| DELAY | 故障切换时重试间隔时间 |

②Server 端 TAF 配置

10gR2 提供 Server 端的 TAF 配置，需要调用 dbms_service 包来在实例上进行修改,例如:

SQL> exec dbms_service.modify_service(service_name => ‘DEMO’,failover_method => 'BASIC',failover_type => 'SELECT',failover_retries => 180,failover_delay => 5);

客户端连接字符串修改成如下即可：

```
demo =
  (DESCRIPTION =
    (ADDRESS=(PROTOCOL=TCP)(HOST=10.194.129.145)(PORT=1521))
   (ADDRESS=(PROTOCOL=TCP)(HOST=10.194.129.146)(PORT=1521))
   (CONNECT_DATA =
      (SERVICE_NAME = demo)
      (SERVER=DEDICATED)
   )
  )
```

(2).FCF 以及实现

FCF 是 10g 引进的一种新的 failover 机制，它依靠各节点的 ons 进程，通过广播 FAN 事件来获得各节点的运行情况，是一种前摄性的判断，支持 JDBC/OCI/ODP.NET

①.利用 onsctl 配置各节点的 local /remote 节点以及端口。配置文件在$ORACLE_HOME/opmn/ons.config

② 配置连接池（以 jdbc 为例）

需要支持 Implicit Connection Cache，设置 FastConnectionFailoverEnabled=true.

将 ojdbc14.jar / ons.jar 等加入环境变量 CLASSPATH.

具体代码可以参见联机文档<Oracle Clusterware and Oracle Real Application Clusters Administration and Deployment Guide>或 metalink 相关文档.

3.Load Balance

10g 的 load balance 同前版本相比有了很大功能上的改进，依据 Load Balancing Advisory，提供了 Runtime Connection Load Balancing 的策略，但个人认为这也是个相对矛盾所在。越是细化的负载均衡机制，越是有加重 cache fusion 的可能，这对 RAC 的整体性能是个考验。

load balance 主要有两种实现方式。一种是 Connection Load Balancing（CLB），另外一种是 Runtime Connection Load Balancing（RCLB）。

CLB 分为客户端 client-side 和服务器端 server-side 两种，client-side 需要在 tnsname.ora 添加 LOAD_BALANCE=ON 来实现，提供基于平衡连接数的负载方案。server-side 需要将修改 remote_listener 参数，让监听器能监听到集群中的所有节点，通过 PMON 来收集节点上的负载信息。

FCF 默认支持 RCLB 功能，RCLB 通过 load balancing advisory 事件来对连接提供更好的服务。RCLB 有两种负载平衡方案可供选择----基于总体 service name 和基于总体 Throughput。可以通过 dbms_service 来设置具体的 goal 方案。

```
SQL> exec dbms_service.modify_service(service_name => 'TEST', goal => DBMS_SERVICE.GOAL_SERVICE_TIME);
```

至于这两种方式的具体差异，在我的测试中，并没有得到显性的体现。

Load Balanc 这部分是我存疑最多的地方，查阅了很多文档，说法不一，且没有翔实的案例证明，在此也希望有过研究的朋友们做指正。

五.其他维护实施相关/案例

本环节侧重一些 rac 施工维护相关的实际案例,对该类施工,从技术角度讲实现起来并无很大难度，在生产施工中，要注意谨慎小心，考虑周全，往往一些细枝末节决定结果和客户的满意度。

1.集群中主机名的更改

2.集群中 IP 地址的更改

3.集群中节点的添加/删除

4.升级:9i rac 升级 10g rac

5.rac + dg 搭建

6.其他

1.集群中主机名的更改

以下是更改前后的主机名称对比,以 hosts 文件为例:

| | 修改前 | 修改后 |
|---|---|---|
| Host Name | node1/node2 | td1/td2 |
| Privite IP Name | node1-priv/node2-priv | td1-priv/td2-priv |
| Vip Name | node1-vip/node2-vip | td1-vip/td2-vip |

(1).生成 listener 的 cap 文件,用于重建 ocr 后注册监听

```
td1-> crs_stat -p
ora.node1.LISTENER_NODE1.lsnr>/home/oracle/ora.node1.LISTENER_NODE1.lsnr.cap
td1-> crs_stat -p
ora.node2.LISTENER_NODE2.lsnr>/home/oracle/ora.node2.LISTENER_NODE2.lsnr.cap
```

(2).停掉所有的资源,备份 ocr、votingdisk 并重新格式化

备份 OCR

```
[root@node1 backup]# ocrcheck
Status of Oracle Cluster Registry is as follows :
        Version                  :          2
        Total space (kbytes)     :     104176
        Used space (kbytes)      :       4424
        Available space (kbytes) :      99752
        ID                       : 2042344313
        Device/File Name         : /dev/raw/raw1
                                   Device/File integrity check succeeded
                                   Device/File not configured
        Cluster registry integrity check succeeded
[root@node1 init.d]# ocrconfig -export /backup/ocr_1.bak
```

备份 votedisk

```
[root@node1 ~]# crsctl query  css votedisk
 0.     0    /dev/raw/raw110
located 1 votedisk(s).
```

重新格式化

```
[root@node1 ~]# dd if=/dev/raw/raw110 of=/backup/votedisk.bak
 [root@td01 ~]# dd if=/dev/zero of=/dev/raw/raw1 bs=1024k count=1
```

```
1+0 records in
1+0 records out
[root@td01 ~]# dd if=/dev/zero of=/dev/raw/raw110 bs=1024k count=1
```

(3).OS 上修改 hostname/hosts 文件，重启主机（步骤略）

(4).重新配置集群互信（步骤略）

(5).编辑$ORA_CRS_HOME/ install/rootconfig 文件，修改以下为你实际的情况

```
ORA_CRS_HOME=/opt/oracle/product/10.2.0/crs_1
CRS_ORACLE_OWNER=oracle
CRS_DBA_GROUP=oinstall
CRS_VNDR_CLUSTER=false
CRS_OCR_LOCATIONS=/dev/raw/raw1
CRS_CLUSTER_NAME=crs
CRS_HOST_NAME_LIST=td1,1,td2,2
CRS_NODE_NAME_LIST=td1,1,td2,2
CRS_PRIVATE_NAME_LIST=td1-priv,1,td2-priv,2
CRS_LANGUAGE_ID='AMERICAN_AMERICA.WE8ISO8859P1'
CRS_VOTING_DISKS=/dev/raw/raw110
CRS_NODELIST=td1,td2
CRS_NODEVIPS='td1/td1-vip/255.255.255.0/eth0,td2/td2-vip/255.255.255.0/eth0'
```

在每个节点依次执行(root 用户)：

```
 [root@td2 install]# /opt/oracle/product/10.2.0/crs_1/install/rootconfig
Checking to see if Oracle CRS stack is already configured
Setting the permissions on OCR backup directory
Setting up NS directories
Oracle Cluster Registry configuration upgraded successfully
WARNING: directory '/opt/oracle/product/10.2.0' is not owned by root
WARNING: directory '/opt/oracle/product' is not owned by root
WARNING: directory '/opt/oracle' is not owned by root
…….
Adding daemons to inittab
Expecting the CRS daemons to be up within 600 seconds.
CSS is active on these nodes.
       td1
       td2
CSS is active on all nodes.
Waiting for the Oracle CRSD and EVMD to start
Oracle CRS stack installed and running under init(1M)
Running vipca(silent) for configuring nodeapps

Creating VIP application resource on (2) nodes...
Creating GSD application resource on (2) nodes...
Creating ONS application resource on (2) nodes...
Starting VIP application resource on (2) nodes...
Starting GSD application resource on (2) nodes...
```

```
Starting ONS application resource on (2) nodes...
```

如果是 10.2.0.1 版本，在最后一个节点上执行的时候会因为静默 vipca 的 bug 抛出异常，在该节点上图形化界面调用 VIPCA。

这样 gsd、ons 和 vip 已经全部注册到 OCR 中。

```
[root@td1 install]# crs_stat -t
Name           Type           Target    State     Host
------------------------------------------------------------
ora.td1.gsd    application    ONLINE    ONLINE    td1
ora.td1.ons    application    ONLINE    ONLINE    td1
ora.td1.vip    application    ONLINE    ONLINE    td1
ora.td2.gsd    application    ONLINE    ONLINE    td2
ora.td2.ons    application    ONLINE    ONLINE    td2
ora.td2.vip    application    ONLINE    ONLINE    td2
```

(6).使用 oifcfg 配置 db 使用的共有、私连网络

```
td1-> oifcfg setif -global eth0/10.194.129.0:public
td1-> oifcfg setif -global eth1/10.10.10.0:cluster_interconnect
```

(7).注册其他资源到集群

①注册监听到集群

修改监听配置文件 lisntener.ora 并编辑生成的 cap 文件（改名），更改其中用到的 hostname.

```
td1-> crs_register ora.td1.LISTENER_TD1.lsnr -dir /home/oracle
td1-> crs_register ora.td2.LISTENER_TD2.lsnr -dir /home/oracle
```

或者使用 netca 图形化界面来配置监听。

②注册 ASM 实例到集群(如果使用 ASM)

```
td1->srvctl add asm -n td1 -i ASM1 -o $ORACLE_HOME
td1->srvctl add asm -n td2 -i ASM2 -o $ORACLE_HOME
```

③注册 instance/database 到集群

```
td1->srvctl add database -d demo -o $ORACLE_HOME
td1->srvctl add instance -d demo -i demo1 -n td1
td1->srvctl add instance -d demo -i demo2 -n td2
```

验证：

```
td1-> crs_stat -t
Name           Type           Target    State     Host
------------------------------------------------------------
ora.demo.db    application    ONLINE    ONLINE    td1
ora....o1.inst application    ONLINE    ONLINE    td1
ora....o2.inst application    ONLINE    ONLINE    td2
ora....SM1.asm application    ONLINE    ONLINE    td1
ora....D1.lsnr application    ONLINE    ONLINE    td1
ora.td1.gsd    application    ONLINE    ONLINE    td1
ora.td1.ons    application    ONLINE    ONLINE    td1
ora.td1.vip    application    ONLINE    ONLINE    td1
ora....SM2.asm application    ONLINE    ONLINE    td2
```

```
ora....D2.lsnr application    ONLINE    ONLINE    td2
ora.td2.gsd    application    ONLINE    ONLINE    td2
ora.td2.ons    application    ONLINE    ONLINE    td2
ora.td2.vip    application    ONLINE    ONLINE    td2
SQL> select * from v$cluster_interconnects;
NAME           IP_ADDRESS      IS_PUBLIC  SOURCE
-------------- --------------- --------- ------------------------------
eth1           10.10.10.145     NO        Oracle Cluster Repository
```

如果使用了 OCFS 作为共享文件格式，注意在启动数据库前检查相应 OCFS 的配置并确认 OCFS 是否能正常挂载使用。

2.集群中 IP 地址的更改

IP 地址的更改比 hostname 的更改相对容易一些。对于同网段的 public/private IP 更改，无需进行特别操作。如果是不同网段，需要使用 oifcfg 处理。因为 VIP 是作为资源注册到 OCR，所以任何 VIP 的改动都需调用相关命令进行处理。

以上处理都需要在集群资源停掉的基础上操作。

以下是修改前后的 public/pricate/vip

| | 修改前 | 修改后 |
|---|---|---|
| Public IP | 10.194.129.145/146 | 10.194.128.145/146 |
| Privite IP | 10.10.10.145/146 | 10.10.1.145/146 |
| Virtual IP | 10.194.129.147/148 | 10.194.128.147/148 |

(1).停掉各资源。

数据库正常关库，其余资源使用 crs_stop 停止。

(2)重新修改网卡 ip/gateway/host 文件等,重启网络等相关服务

(3).利用 oifcfg 更改 public/private ip

查看当前使用

```
td1-> oifcfg getif
eth1  10.10.10.0  global  cluster_interconnect
eth0  10.194.129.0  global  public
```

删除当前

```
td1-> oifcfg delif -global eth0
td1-> oifcfg delif -global eth1
```

重新添加

```
td1-> oifcfg setif -global eth0/10.194.128.0:public
td1-> oifcfg setif -global eth1/10.10.1.0:cluster_interconnect
```

(4).更新注册到 OCR 中的 vip 配置(root 用户)

```
[root@td1 ~]# crs_register -update ora.td1.vip -o
oi=eth0,ov=10.194.128.147,on=255.255.255.0
[root@td1 ~]# crs_register -update ora.td2.vip -o
oi=eth0,ov=10.194.128.148,on=255.255.255.0
```

或者使用(root 用户)

```
[root@td1 ~]#  srvctl modify nodeapps -n td1 -A 10.194.128.147/255.255.255.0/eth0
[root@td1 ~]#  srvctl modify nodeapps -n td2 -A 10.194.128.148/255.255.255.0/eth0
```

(5).如果你使用了 ocfs，修改 ocfs 配置文件(/etc/ocfs/cluster.conf),验证修改后是否可用。

(6).修改监听 listener 配置文件

(7)启动集群各节点资源并验证

```
td1-> crs_start –all
```

登陆数据库，检验内连接是否生效。

```
SQL> select * from v$cluster_interconnects;
NAME            IP_ADDRESS      IS_PUBLIC SOURCE
--------------- --------------- --------- ---------------------------
eth1            10.10.1.145     NO        Oracle Cluster Repository
```

### 3.集群中节点的删除/添加

同 9i 的节点删除/添加相比，10g 对节点的添加和删除相对来说略显麻烦，但操作起来更加规范。

因为集群件的存在，需调用一系列接口命令将资源从 OCR 中添加/删除，本文不再对该案例做详细描述，具体参见 oracle 官方联机文档 RAC 部分<Adding and Deleting Nodes and Instances on UNIX-Based Systems>.

### 4.升级与迁移

RAC 的迁移与升级并不比单实例复杂多少。对于一个 rac 新手来说，在思想上也无需觉得这是个很庞杂的事情，当然前提是你有足够的单实例方面的基础知识并对此深刻理解。

比如，利用 rman 备份，我们可以方便的将一个运行在单节点的实例迁移到 rac 环境下。需要做的重点，仅仅是迁移数据库(你可以想象成是单实例对单实例)，然后编辑参数文件，添加其他节点启动 db 所必要的 redo 和 undo，并注册数据库资源到集群中以便管理。

如果你的迁移或升级有停机时间的限制，那大部分情况下重点的问题并不在于被操作对象是 RAC 架构，而在于如何制定你的 MAA 策略。比如你需要运用一些表空间传输或者高级复制/流等方面的特性来压缩停机时间，这也许因为是 RAC 架构而增加了整个施工的难度，但很多时候问题的重点并不在于此。

接下来提供一个 9I RAC 同机静默升级到 10G RAC 的案例，我并没有整理到该手记中。详细可参见我的 blog http://www.easyora.net/blog/9i_rac_upgrade_10g_rac.html

### 5.高可用架构:RAC+DG

应该说，rac+dg 企业级的架构在高可用和灾备方面来说还是有相当大的市场。

在搭建与管理方面，rac(主)+DG(备)的过程与单实例的主备也无太大异同。需要注意以下方面(但不限于以下方面):

(1).log gap 的检测问题

注意正确配置 fal_server 与 fal_clicent 参数，尤其是对于 rac 主库归档到各自节点上的情况下,standby 端 gap server 需要将每个主库节点都涵盖进去。

(2)switchover/failover 注意事项

做任何切换的时候，需要将 rac 端只保留一个 alive 的实例，其余实例关闭后，切换步骤跟单节点 dg 基本一致。

(3) standby logfile 问题

如果采用 LGWR 传输日志，需要备库端添加 standby logfile 日志。需要注意添加的 standby logfile 的 thread 要与主库一致。如果你的主库节点有 3 个实例，那需要添加 3 大组与 rac 主库相同 thread 号的备用日志，每个 thread 至少 2 组日志即可。

六．RAC 监控优化

### 1.思路及等待事件说明

鉴于 RAC 体系的复杂性，RAC 的优化比单实例的优化给我们提出了更高的难度和要求。大部分情况下，单实例上的优化方法在 RAC 结构下同样适用。

RAC 优化的 2 个核心问题:

(1).减少 shared pool 的压力:减少对数据字典的争用，减少硬解析。

因为 row cache/library cache 是全局的，频繁的数据字典争用/硬解析在 RAC 环境下会造成比单实例更严重的性能后果。

(2). 减少因 Cache fusion 带来的全局块传输和争用

频繁的 Cache fusion 会带来一系列数据块上的全局争用。如何减少逻辑读，尽量减少数据在实例之间共享传输，是 RAC 体系对应用设计和部署的新要求

Cache fusion 性能是影响 RAC 系统性能的一个极为重要的方面。Avg global cache cr block receive time 和 avg global cache current block receive time 是 cache fusion 的两个重要指标，以下是 oracle 给出的这两个指标的阈值情况:

| Name | Lower Bound | Typical | Upper Bound |
|---|---|---|---|
| Avg global cache cr block receive time(ms) | 0.3 | 4 | 12 |
| Avg global cache current block receive time(ms) | 0.3 | 8 | 30 |

RAC 下的全局等待事件:

```
SQL>select * from v$event_name  where NAME like 'gc%' and WAIT_CLASS='Cluster';
```

10G R2 下有 40 多个非空闲的全局等待时间，最常见的值得引起注意的等待事件如下：

**gc current/cr request**

该等待事件表示资源从远程实例读取到本地实例所花费的时间。出现该事件并不能说明什么问题，如果等待时间过长，可能表示内联网络存在问题或者有严重的块争用。

**gc buffer busy:**

buffer busy waits 在全局上的延伸。出现该等待时间一般可能是块的争用问题。

**Enquenue**

RAC 中，常见的特别 Enquenue 有 enq: HW – contention/ enq: TX - index contention/enq 等，在跨节点高并发的 insert 环境中很容易出现

诸如 gc current-2way/3way.gc current/cr grant 等事件，这些事件只是提供了块传输和消息传输方面的细节或是结果，一般情况下无需太投入关注。

### 2.性能诊断

性能上的调整很难给出一个定式，但指导思想上可以实现很大方面的统一。

AWR/ASH 等报告可以作为 RAC 系统中一个强有力的性能采集和诊断工具。同单实例的报告相比，AWR 中的 RAC Statistics 部分给我们提供了详细的 GES、GCS 性能采样，结合全局等待事件，定位集群问题上的症状。

在 RAC 结构下，Segment Statistics 部分是我们更加需要注意的地方。如果你还是习惯使用 STATSPACK 来进行性能采集，建议至少将收集级别设置为 7。该部分为我们提供了详细的 Segment 级别的活动情况，有助于我们定位全局的 HOT table /HOT index，分析全局资源排队争用的根源。

要重视 DBA_HIS 开头的一系列视图的作用，这将帮我们将问题定位的更加细化，甚至定位到 SQL 级别。糟糕的 SQL 效率拖垮系统性能的案例比比皆是，这在 RAC 中往往更加常见。dba_hist_active_sess_history 可以作为很好的切入点，例如通过关联 dba_hist_sqltext 获得执行文本，通过关联 dba_hist_sql_plan 获得执行计划树等，有时候将直接找到造成等待事件的元凶。

RAC 中常见的争用和解决方法：

① Sequence and index contention

Sequence 是 RAC 中容易引起争用的一个地方,尤其是以 sequence 作索引，在高并发的多节点 insert 情况下极易引起索引块的争用以及 CR 副本的跨实例传输。

需要尽量增大 Sequence 的 cache 值并设置序列为 noorder。

② undo block considerations

RAC 下 CR 的构造比单实例成本要高，如果一个 block 中的活动事务分布在几个实例上，需要将几个实例上的 undo 合并构造所需要的 CR，尤其是高并发的有索引键的插入，容易造成 undo block 的争用。

尽量使用小事务处理。

③HW considerations

跨节点的高并发 insert 会造成高水位线的争用，采用更大的 extent/采用 ASSM 和分区技术能减缓这一争用。

③ Hot Block

全局热点块问题对 RAC 系统的影响巨大，尽量减少块跨实例的并发更改,适当采用分区可以缓解该争用。

一个良好的应用设计是 RAC 发挥功力的重要前提，根据不同的节点部署不同的应用，能有效的减少全局资源的争用，对 RAC 性能的稳定也相当重要。

参考:

1. Oracle 10g RAC Grid,services & Clustering --Murali Vallath
2. Oracle Database 10g 高可用性实现方案——运用 RAC、Flashback 和 Data Guard 技术 --Matthew Hart,Scott Jesse
3. Oracle Wait Interface: A Practical Guide to Performance Diagnostics & Tuning –Richmond Shee
4. Oracle 官方联机文档 --otn.oracle.com