ORACLE 11gR2 RAC Architecture

Author	**박제헌,신범석**
Creation Date	**2011-03-02**
Last Updated	
Version	**1.0**

Version	변경일자	변경자(작성자)	주요내용
1	2011-03-02	박제헌,신범석	문서 최초 작성

Goodus
Breathing Life into Technology

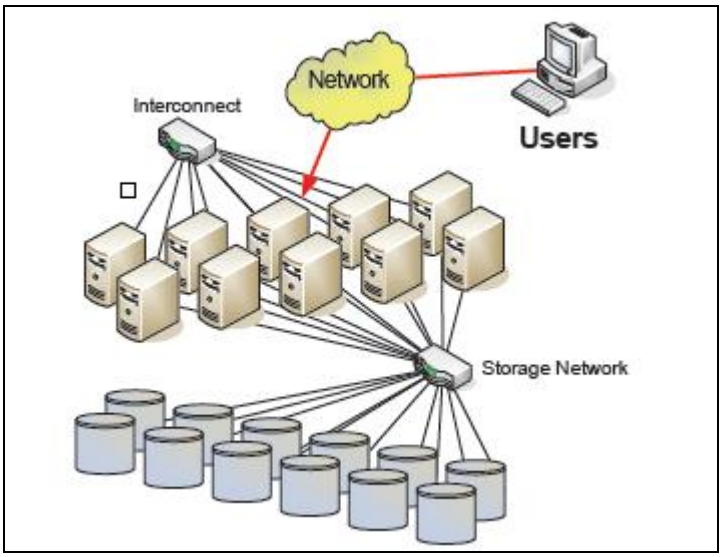
목 차

이번 기술노트에서는 11gR2 의 아키텍처에 대해서 알아본다. 11g 클러스터의 개념에 대하여 알아보고 클러스터의 아키텍처와 ASM 의 아키텍처에 대해 알아본다. 11gR2 부터 rawdevice 가 공식적으로 지원 되지 않고 ASM 사용을 권장하므로 ASM 의 아키텍처와 새로운 기능에 대하여 알아보자.

1. Oracle Clusterware Concepts

1.1. What Is a Cluster?

- 단일 서버로 작동하는 상호 연결된 노드
- 클러스트 소프트웨어는 구조를 숨긴다.
- 모든 노드에서 디스크를 읽고 디스크에 쓸 수 있다.
- 운영 체제는 모든 컴퓨터에서 동일 하다



클러스터는 둘 이상의 독립적이지만 상호 연결된 서버로 구성된다. 하드웨어 업체에서는 다양한 요구 사항을 충족시키기 위해 클러스터 기능을 제공합니다. Active node 가 실패하면 보조 노드로 작업을 전송할 수 있도록 허용함으로써 높은 가용성을 제공하는 용도로만 사용되는 클러스터도 있고, 노드 간의 유저 연결 또는 작업 분산을 허용함으로써 확장성을 제공하도록 설계된 클러스터도 있다.

클러스터의 공통된 특징은 응용 프로그램에서 단일 서버처럼 표시 된다는 것입니다. 마찬가지로 여러 서버의 관리 작업도 최대한 단일 서버의 관리 작업과 비슷해야 하며 클러스터 관리 소프트웨어는 이러한 투명성을 제공한다.

여러 노드가 단일 서버처럼 작동하려면 파일을 필요로 하는 특정 노드에서 찾을 수 있도록 저장해야 한다. 서로 다른 여러 클러스터 토폴로지에서 데이터 액세스 문제를 해결하며, 각 토폴로지는 클러스터 디자이너의 기본적인 목표에 따라 달라진다.

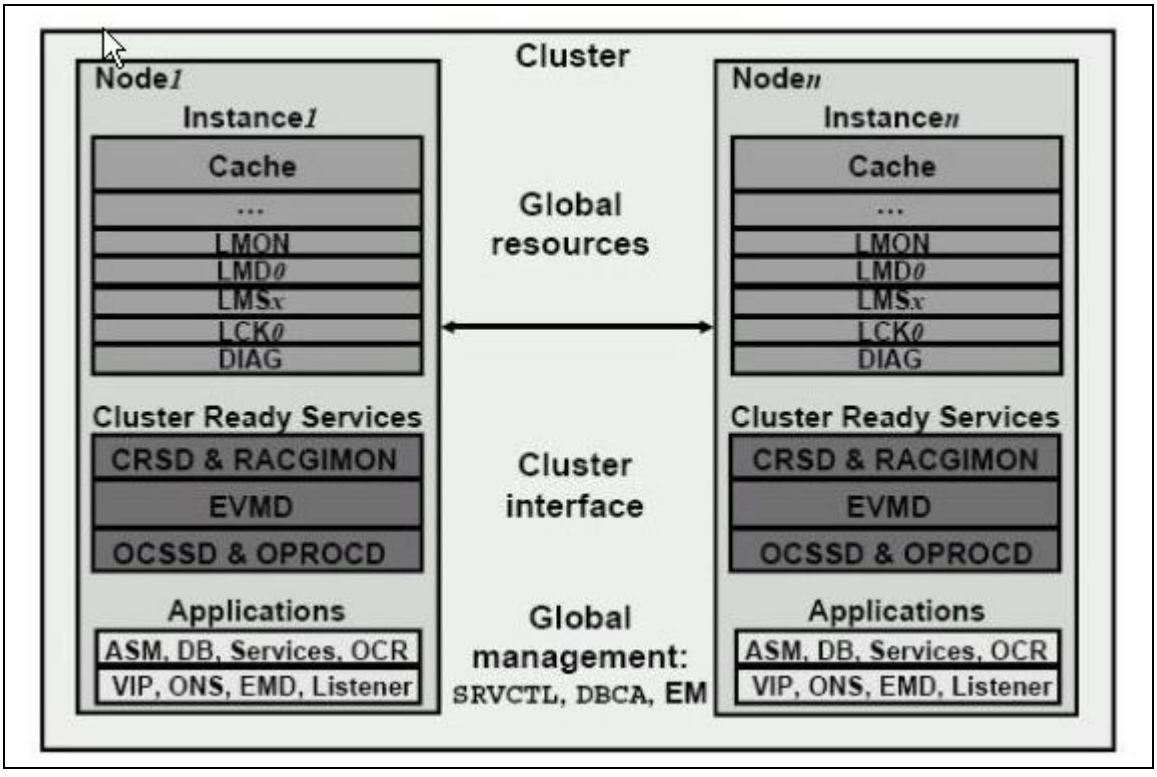
상호 연결은 클러스터의 각 노드 간에 통신 수단으로 사용되는 물리적 네트워크이다.
요약하자면 클러스터는 단일 시스템으로 통합되는 개별 서버의 그룹이다.

1.2. Oracle Real Application Cluster

- 여러 Instance 가 동일한 데이터베이스에 액세스
- 노드당 하나의 Instance
- 각 데이터베이스 파일에 대한 물리적 또는 논리적 액세스
- 소프트웨어가 제어하는 데이터 액세스

Real Application Clusters 는 동일한 데이터베이스에 대해 여러 Instance 를 실행하여 클러스터된 하드웨어를 사용할 수 있도록 하는 소프트웨어입니다. 데이터베이스 파일은 각 노드에 물리적 또는 논리적으로 연결된 디스크에 저장되므로 모든 활성 Instance 에서 읽고 쓸 수 있다.
Real Application Clusters 소프트웨어는 데이터 액세스를 관리하여 Instance 간에 변경사항을 조정하며, 각 Instance 에 일관성 있는 데이터베이스 이미지가 표시되도록 합니다. 그리고 클러스터 상호 연결을 통해 Instance 가 서로 조정 정보와 데이터 이미지를 전달할 수 있다.
RAC 구조는 시스템 작동 중지나 사용 불능 상태 등의 경우에 중복성을 제공하므로 응용프로그램이 정상적인 Instance 에서 데이터베이스에 계속 액세스할 수 있다.

1.3. RAC 소프트웨어 원리



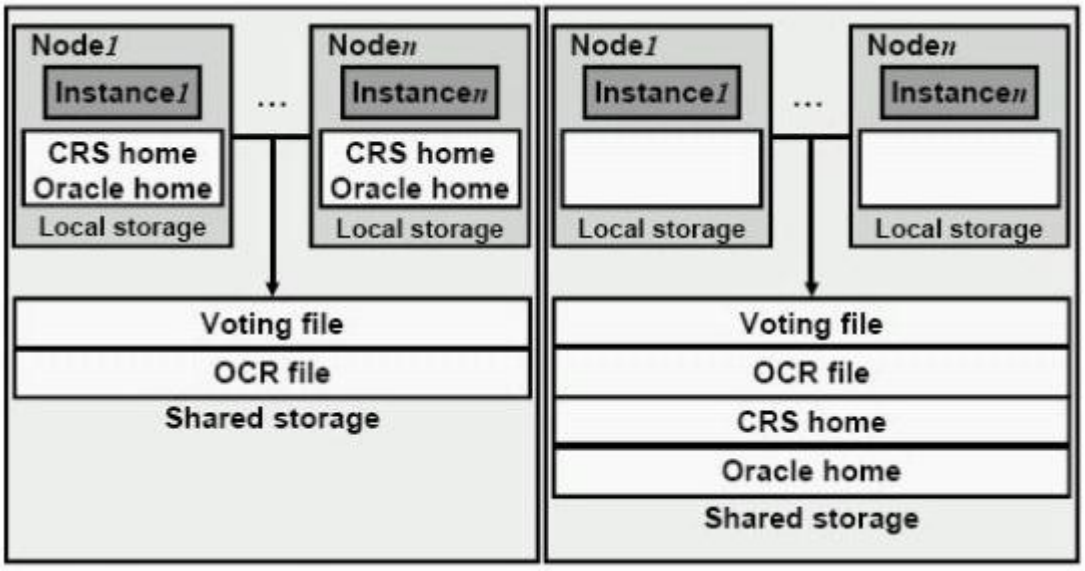
RAC Instance 에는 단일 Instance 데이터베이스에 비해 적은 추가 백그라운드 프로세스가 연결 되어 있다. 이러한 프로세스는 주로 각 Instance 간의 데이터베이스 일관성을 유지하는데 사용된다.

- LMON : Global Enqueue Service 모니터
- LMD0 : Global Enqueue Service Daemon
- LMSx : Global Cache Service 프로세스(x 의 범위는 0 에서 j 사이)
- LCK0 : Lock 프로세스
- DIAG : 진단 가능성 프로세스

클러스터 레벨에서는 Oracle Clusterware 의 기본 프로세스가 있다. 이들 프로세스는 모든 플래폼에서 표준 클러스터 인터페이스를 제공하며 가용성이 높은 작업을 수행한다. 이러한 프로세스는 클러스터의 각 노드에 있다.

- CRSD 및 RACGIMON : 고가용성 작업용 엔진
- OCSSD : 노드 멤버 및 그룹 서비스에 대한 액세스 제공
- EVMD : 콜 아웃 디렉토리 스캔 및 감지된 이벤트에 대한 응답으로 콜 아웃 호출
- OPROCD : 클러스터용 프로세스 모니터

1.4. RAC 소프트웨어 저장 영역 원리



패치 업그레이드 롤링 허용
소프트웨어는 단일 Failure 지점이 아님

Real Application Clusters 는 두 단계를 수행하여 설치합니다. 첫번째 단계에서는 Oracle Clusterware 를 설치하고, 두번째 단계에서는 RAC 구성 요소를 포함하는 오라클 데이터베이스 소프트웨어를 설치한 다음 클러스터 데이터베이스를 생성한다. Oracle Clusterware 에 사용하는

Oracle Home 은 RAC 소프트웨어에 대해 사용하는 Oracle Home 과는 달라야한다. 특정 클러스터 파일 시스템을 사용할 때는 클러스터 공유 저장 영역에 RAC 소프트웨어를 설치할 수 있지만, 소프트웨어는 보통 각 노드의 로컬 위치인 일반 파일 시스템에 설치 된다. 이로 인해 패치 업그레이드를 롤링할 수 있으며 소프트웨어가 단일 Failure 지점이 되지 않는다. 또한 공유 저장 영역에는 최소한 다음과 같은 두 개의 파일을 저장해야 한다.

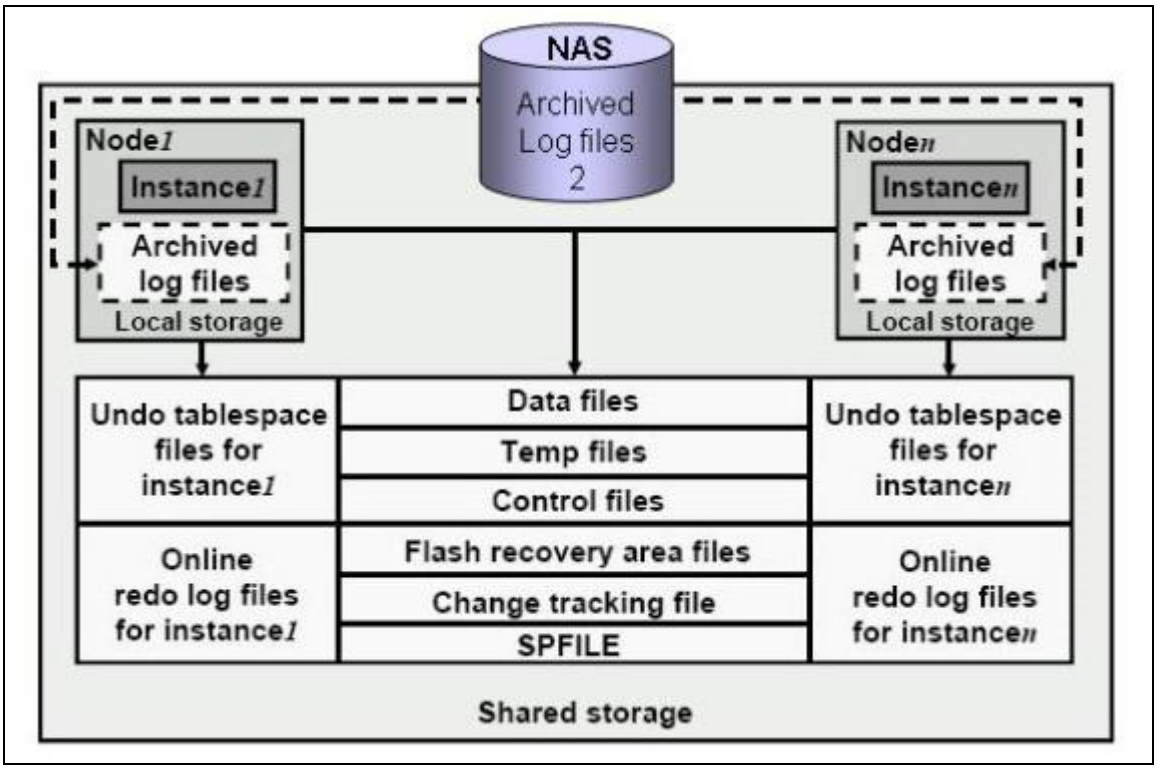
- Voting file
  클러스터에 대한 노드 감시 정보를 위해 CSSD(Cluster Synchonization Services Daemon)에 의해 사용되는 필수적인 file.

- OCR(Oracle Cluster Registry)
  CRS 의 핵심 구성요소이며 클러스터의 HA 컴포넌트의 정보를 관리.

- Cluster node list
- Cluster Database Instance to node mapping
- CRS Application resource profile(service, VIP)등
- Size : 100MB 이상

Voting 파일 및 OCR 파일은 오라클 Instance 를 시작하기 전에 액세스해야 하므로 ASM 에 저장할 수 없다. OCR 파일과 Voting 파일은 RAID 등의 안정적인 중복 저장 영역에 저장할 수도 있고 서로 다른 디스크에 Mirroring 할 수도 있다. 가장 적합한 위치는 속도가 빠른 I/O 디스크의 Raw Device 이다.

1.5. RAC 데이터베이스 저장 영역 원리



RAC 저장 영역과 단일 Instance 오라클 데이터베이스용 저장 영역의 주된 차이점은 RAC 의 모든 데이터 파일은 동일한 데이터베이스에 액세스하는 모든 Instance 에서 공유할 수 있도록 공유장치(Raw Device or 클러스터 파일 시스템)에 상주해야 한다는 점이다. 또한 각 Itance 에 대해 두개 이상의 리두로그 그룹을 생성해야 하며, 모든 리두 로그 그룹은 Instance 또는 Crash Recovery 에 사용할 수 있도록 공유 장치에 저장해야 한다. 각 Instance 의 온라인 리두 로그 그룹을 Instance 의 온라인 리두 스레드라고 한다.

권장되는 AUM(Automatic Undo Management)를 사용하려면 각 Instance 에 대해 하나의 공유 언두 테이블스페이스를 생성해야 한다. 각 Instance 의 언두 테이블스페이스는 Recovery 용으로 다른 모든 Instance 와 공유해야 한다.

아카이브 로그는 각 로그마다 다른 이름이 자동으로 생성되기 때문에 Raw Device 에 배치할 수 없다. 그러므로 아카이브 로그는 파일 시스템에 저장해야 한다. CFS(Cluster File System)를 사용하는 경우에는 해당 시스템을 통해 원하는 노드에서 이러한 하카이브 파일에 액세스할 수 있다. CFS 를 사용하지 않는 경우에는 Recovery 를 수행할 때 항상 아카이브를 다른 클러스터 멤버가 사용할 수 있도록 해야한다.(예: 노드 전체에 걸쳐 NFS 사용) Flash Recover Area 를 사용하는 경우에는 모든 Instance 에서 액세스할 수 있도록 아카이브를 공유 디렉토리에 저장해야 한다.

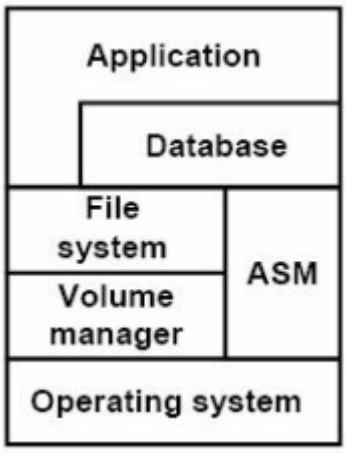
1.6. RAC 및 공유 저장 영역 기술

- 저장 영역은 그리드의 핵심 구성 요소
  - 저장 영역 공유는 기본적인 기능
  - 새로운 기술이 도입되고 있는 추세
- Oracle Grid 에 대해 지원되는 공유 저장 영역
  - NAS(Network Attached Storage)
  - SAN(Storage Area Network)
- Oracle Grid 에 대해 지원되는 파일 저장 영역
  - Raw 볼륨
  - 클러스터 파일 시스템
  - ASM

RAC 에 필요한 공유 저장 영역을 제공하기 위해 기본적으로 세 가지 접근 방식이 사용된다.

- Raw 볼륨 : 파이버 채널 또는 iSCSI 등 블록 모드에서 작동하는 저장 영역을 필요로 하는 직접 연결된 Raw Device.
- 클러스터 파일 시스템 : 하나 이상의 클러스터 파일 시스템을 사용하여 모든 RAC 파일을 보관할 수 있다. 클러스터 파일 시슽ㅁ을 사용하려면 파이버 채널 또는 iSCSI 등의 블록 모드 저장 영역이 필요하다.
- ASM
  - 이동 가능하고 고성능의 Cluster File System
  - Oracle Database files 를 관리
  - 부하 균형을 위해 디스크에 데이터 분산

1.7. 자동 저장 영역 관리(ASM)



- 이동가능한 고성능 데이터베이스 파일 시스템 제공
- 로드가 균형을 이루도록 여러 디스크에 데이터 분산
- 전체 디스크에 걸쳐 통합된 Mirroring 제공
- 많은 저장 영역 관리 문제 해결

ASM 은 Oracle Database 10g 부터 제공되는 기능으로, 오라클 데이터베이스 파일용으로 특별히 구축된 볼륨 관리자와 파일 시스템을 종적으로 통합한 것이다. ASM 은 단일 시스템의 관리

기능이나 Oracle RAC(Real Application Clusters)지원을 위해 클러스터의 노드 간에 관리 기능을 제공한다.

- ASM 은 사용 가능한 모든 리소스에 I/O 로드를 분산하여 성능을 최적화하면서 수동 I/O 튜닝의 필요성도 제거한다. ASM 은 데이터베이스를 종료하지 않고 데이터베이스 크기를 늘려 저장 영역을 늘릴수 있도록 동적 데이터베이스 환경을 관리 할 수 있다.
- ASM 은 중복 데이터 복사본을 유지 관리하여 결함 허용 기능을 제공할 수있어 안정적인 저장 영역 메커니즘을 제공한다.
- ASM 기능을 사용하면 수동 저장 영역을 자동화할 수 있으므로 효율적으로 관리할 수 있다.

1.8. CFS와 Raw

- CFS 사용시 이점
  - 보다 손쉬운 관리
  - RAC 를 통한 OMF(Oracle Managed Files) 사용
  - 단일 Oracle 소프트웨어 설치
  - Oracle 데이터 파일에서 Autoextend 사용 가능
  - 물리적 노드 Failure 가 발생하는 경우에도 아카이브 로그에 동일하게 액세스 가능
- Raw 사용의 이점
  - 높은 성능을 제공
  - CFS 를 사용할 수 없을때도 사용가능
  - 아카이브 로그 파일에는 사용 불가
  - 손쉬운 작업 수행

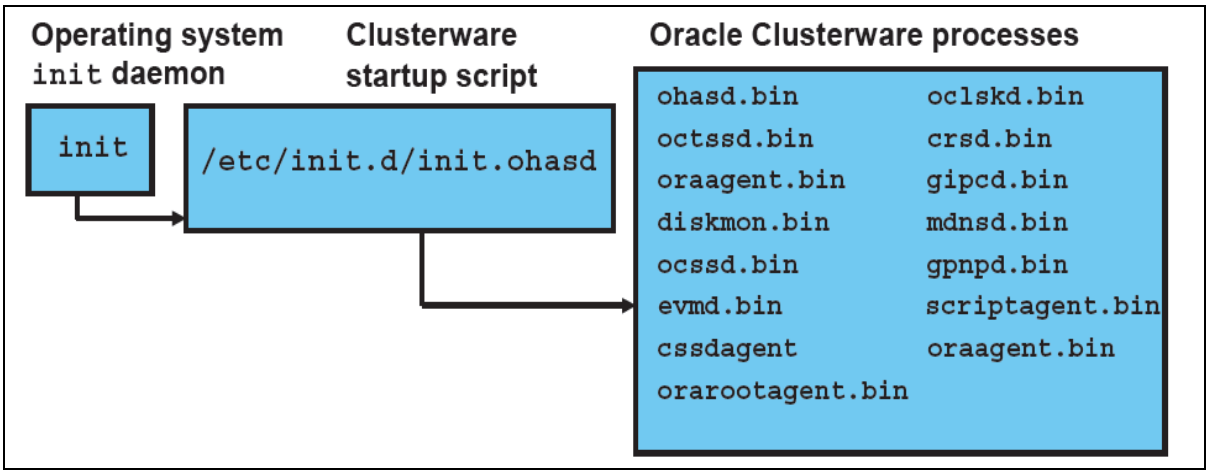
2. Clusterware Architecture

2.1. Oracle Cluster Damon

오라클 클러스터 웨어는 OS init 데몬에 의해서 시작된다.
Init.ohasd 스크립트가 오라클 클러스터 프로세서들을 시작시키는데 이 스크립트는 /etc/init.d 에 카피된다.

```
# cat /etc/inittab
..
h1:35:respawn:/etc/init.d/init.ohasd run >/dev/null 2>&1 </dev/null
```

Init.ohasd 에 의해 시작되는 오라클 클러스터 프로세서에 대해 알아보자.



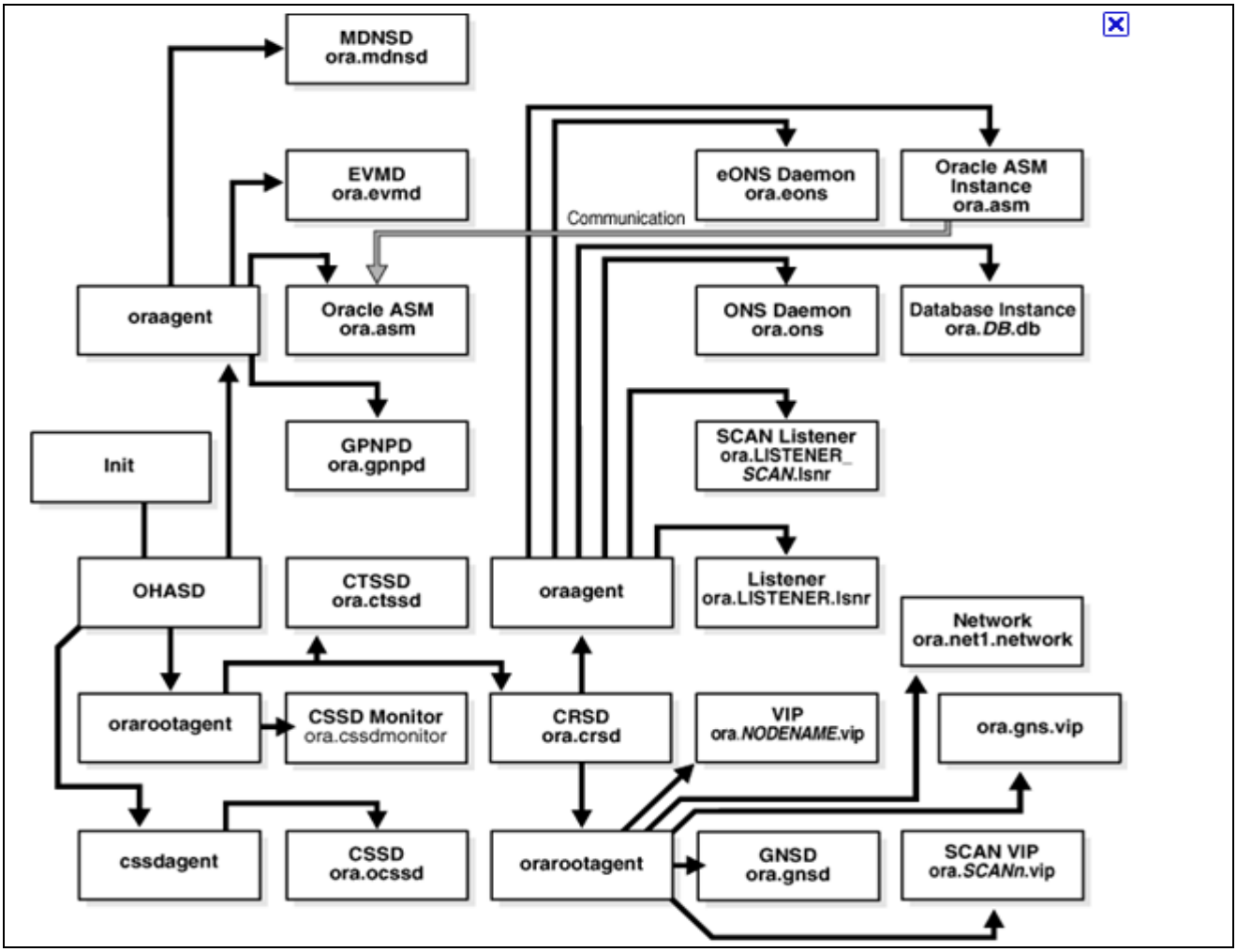
crs 관련 데몬과 시작되는 프로세스.

Daemon	Start_process
Ohasd(Oracle High Availability Services Daemon) 오라클 관련 데몬들을 모니터링, 재시작	orarootagent, cssdagent, oraagent
Orarootagent	crsd, root-owned CRS resources(scan vips)
Cssdagent	cssd(ocssd)
oraagent	mdnsd, evmd, ASM, ctssd, gpnpd gsd ,ONS,listener

오라클 클러스터 프로세스 아키텍쳐

Component	Processes	Owner
Cluster Ready Service(CRS)	crsd	root
Cluster Synchronization Service(CSS)	ocssd, cssdmonitor, cssdagent	grid owner, root
Event Manager (EVM)	evmd, evmlogger	grid owner
Cluster Time Synchronization Service (CTSS)	octssd	root
Oracle Notification Service (ONS)	ons, eons	grid owner
Oracle Agent	oraagent	grid owner
Oracle Root Agent	oraootagent	root
Grid Naming Service(GNS)	gnsd	root
Grid Plug and Play (GPnP)	gpnpd	grid owner
Multicast domain name service(mDNS)	mdnsd	grid owner

CRS 관련데몬이 시작되는 과정을 그림으로 보면 아래와 같다.



CRS (Cluster Ready Service)

클러스터에 High availability 운영을 위한 주 프로그램이다.

CRS daemon (crsd)는 OCR 에 저장된 클러스터 리소스를 관리한다.시작,정지,모니터,패일오버 등의 오퍼레이션이 포함되어 있다. Cssd 프로세스는 리소스의 변화가 있을 때 동작한다. 오라클 데이터 베이스 인스턴스 ,리스너 프로세스가 실패시에 자동으로 재시작하는 경우가 이런 경우라 할수 있다.

CSS (Cluster Synchronization Services)

클러스터 노드의 멤버들을 관리한다. 노드가 클러스터에 추가되거나 분리되었을 때다른 멤버들에게 알리고 클러스터의 구성을 컨트롤 한다.

CSS 는 CSS daemon(ocssd), CSS Agent(cssdagent), CSS Monitor(cssdmonitor) 세 가지 분리된 프로세스가 있다.

Cssdagent process 는 클러스터 모니터와 I/O fencing 을 제공한다.

11.2.0.2 이전 버전 까지는 I/O fencing 을 oprocd 가 담당하게 되는데 이 프로세스는 시스템이 타 벤더 클러스터를 사용하지 않을 때 생성되는 프로세스이다. 11.2.0.2 버젼부터 I/O fencing 을 cssadagent 가 담당하게 된다.

> * I/O fencing
> 여러 노드가 공유 스토리지의 데이터를 동시에 접근할 때 노드간의 통신을 통해 스토리지의 사용 여부를 체크하면서 서로 데이터를 쓰게 되는 것을 방지함으로 데이터 무결성을 보장할 수 있는데, 노드간의 커뮤니케이션이 실패하였을 때 split brain 현상이 발생한다. Split brain 은 각 서버가 각각 스토리지를 컨트롤 하게되는 현상인데 이런 현상으로 인한 데이터 손상의 방지와 무결성을 위해 I/O fencing 을 사용하게 된다.

Cssdagent 는 오라클 클러스터가 fail 되었을 때 노드를 재부팅 시키고, interconnector 가 fail 되었을 시에도 노드를 재부팅 시킨다. 11.2.0.2 부터는 노드 재부팅의 횟수를 줄이고자 새로운 메커니즘을 적용하였다. 노드 재 부팅을 하는 대신 문제가 되는 프로세스 kill 을 시도하게 된다. Cssd 가 문제의 프로세스가 kill 을 하고 kill 이 잘 되었는지 확인을 하는 과정에서 kill 이 정상적으로 되었을 경우에는 OHASD 가 CRS 만을 재 시작하게 되고 kill 을 실패하게 되었을 때에는 CRS 를 재 시작 하는 것이 아니라 노드를 재부팅 시킨다.

EVM (Event Manager)

오라클 클러스터에서 발생되는 이벤트를 publish 하는 백그라운드 프로세스이다.

CTSS (Cluster Time Synchronization Service)

오라클 클러스터의 시간을 관리하는 프로세스이다.기존에는 NTP(Network Time

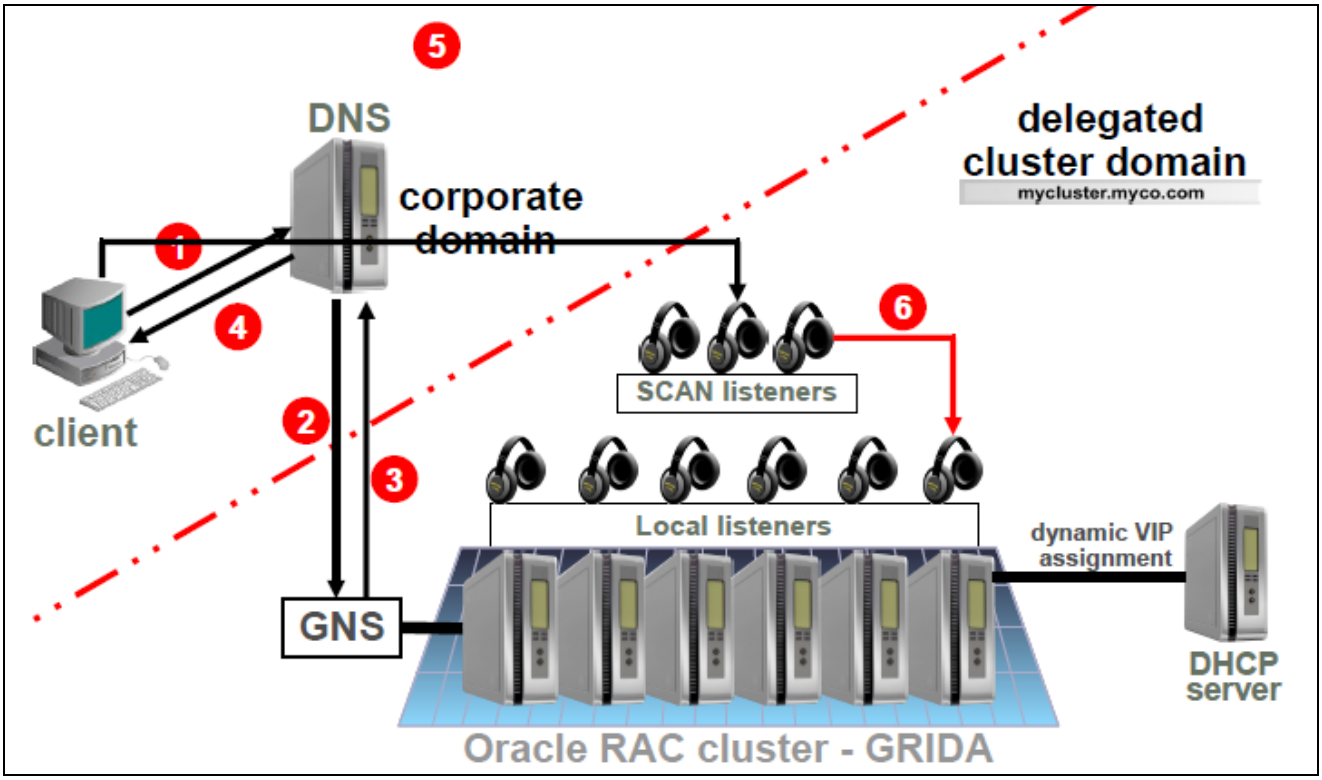
Protocal)을 이용하여 시간 동기화를 하였지만 11gR2 부터는 CTTS 를 이용하여 클러스터의 시간을 동기화 한다.

ONS (Oracle Notification Service)

FAN(Fast Application Notification) event 가 발생 시에 ONS 를 통해 event 를 받게 된다. 데이터베이스 나 인스턴스가 시작,정지 시에 FAN 이벤트가 발생하게 되는데 이때 데이터베이스나 인스턴스의 시작,정지 등을 인지할 수 있다. ($ORA_CRS_HOME/racg/usrco 에 스크립트를 만들어 지정)

2.2. GNS (Oracle Grid Naming Service)

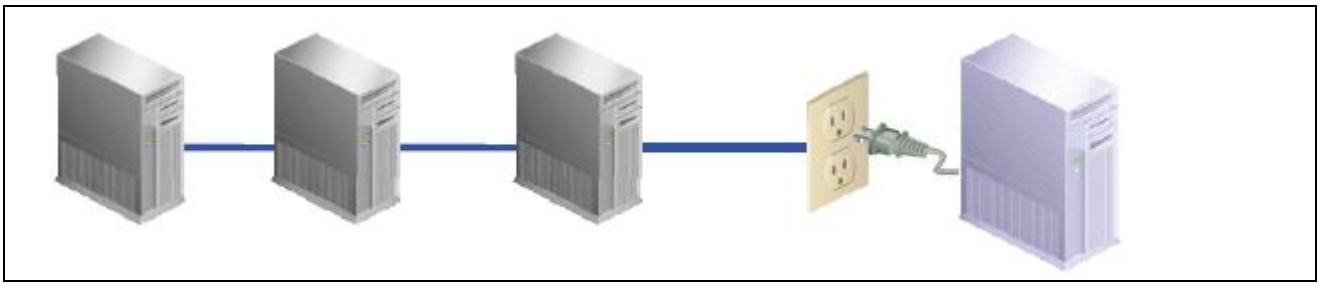
클러스터 DNS 와 external DNS 서버간의 게이트 웨이 역할을 하며, 클러스터 내의 이름 resolution 을 수행한다. VIPs, SCAN-VIP 가 GNS 에 정의 되어있고, GPnP 사용시에 구성되어야 한다. GNS 를 위한 도메인(grid.example.com) 과 IP 가 필요하다.

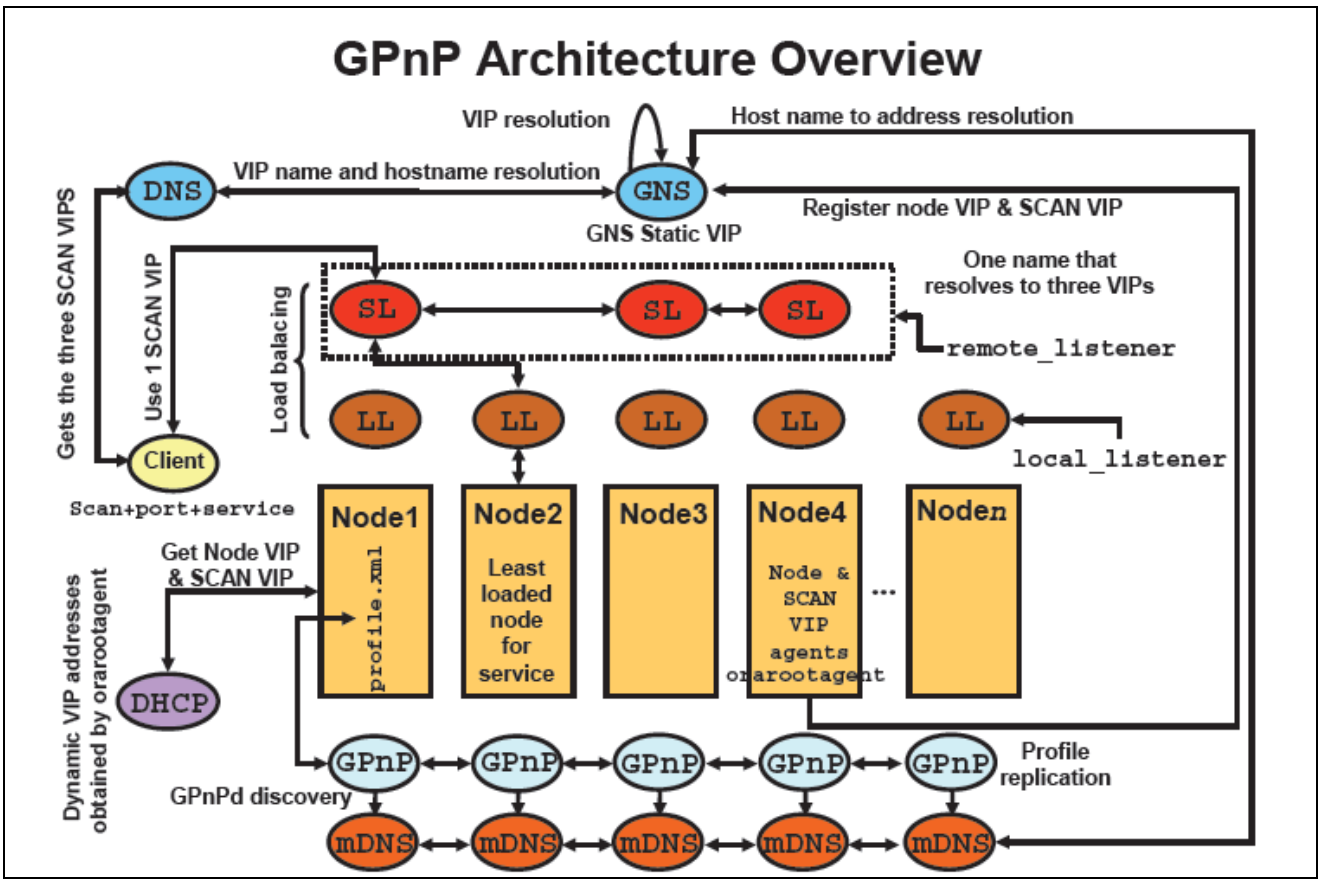


2.3. Grid Plug and Play

GPnP 를 이용하여 클러스터에 노드를 쉽게 추가,삭제를 할 수 있으며 virtual ip addresses 를 관리한다.

이전 버전에서 서버를 클러스터에 추가하기 위해서는 수동으로 작업을 해야 했지만, 11gR2 버젼부터는 수동 으로 서버를 클러스터에 추가할 수 있을 뿐아니라 Grid Plug and play 를 통해 동적으로 노드를 추가 , 제거 구성의 가능하다.





GPnP 는 노드에 대한 profile 을 가지고 있는데 노드가 추가시 이 profile 이 복제된다

Profiles.xml

```
$ cat GRID_HOME/gpnp/profiles/peer/profiles.xml
<?xml version="1.0" encoding="UTF-8"?><gpnp:GPnP-Profile Version="1.0"
```

```
xmlns="http://www.grid-pnp.org/2005/11/gpnp-profile" ...
xsi:schemaLocation="http://www.grid-pnp.org/2005/11/gpnp-profile gpnp-profile.xsd"
ProfileSequence="4" ClusterUId="2deb88730e0b5f1bffc9682556bd548e"
ClusterName="cluster01"
PALocation=""><gpnp:Network-Profile><gpnp:HostNetwork id="gen"
HostName="*"><gpnp:Network
id="net1" IP="192.0.2.0" Adapter="eth0" Use="public"/><gpnp:Network id="net2"
IP="192.168.1.0" Adapter="eth1"
Use="cluster_interconnect"/></gpnp:HostNetwork></gpnp:Network-
Profile><orcl:CSS-Profile
id="css" DiscoveryString="+asm" LeaseDuration="400"/><orcl:ASM-Profile id="asm"
DiscoveryString="/dev/sd*" SPFile="+data/spfile.ora"/><ds:Signature
<ds:SignedInfo>
<ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-
c14n#"/>
<ds:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
<ds:Reference URI="">
<ds:Transforms>
<ds:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
signature"/>
<ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#">
<InclusiveNamespaces xmlns="http://www.w3.org/2001/10/xml-exc-c14n#"
PrefixList="gpnp orcl xsi"/>
...
<ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
<ds:DigestValue>gIBakmtUNi9EVW/XQoE1mym3Bnw=</ds:DigestValue>
```

mDNS (Multicast domain name service)

DNS 로의 requests 를 허용하는 백그라운드 프로세스이다.

2.4. SCAN (Single Client Access Name)

2.4.1. SCAN(Single Client Access Name)

클라이언트에서 싱글네임으로 클러스터안의 어느 데이터 베이스나 서비스에 접속할 수 있는 11gR2 RAC 의 새로운 기능이다. SCAN IP 가 싱글네임 정보와 함께 GNS 에 등록되어 있으므로 싱글네임으로 데이터 베이스에 접속 시 클러스터에 노드가 추가나 삭제가 되어도 수정이 필요 없으며 간단한 JDBC thin URL 로 접속이 가능하다. 클라이언트 접속 시 로드 발랜싱과 패일 오버를 지원한다. Tnsnames.ora 의 예전버전과 비교하여 알아보자

예전 버전의 tnsnames.ora

```
        RAC =
          (DESCRIPTION =
            (ADDRESS = (PROTOCOL = TCP)(HOST = rac1_vip)(PORT = 1521))
            (ADDRESS = (PROTOCOL = TCP)(HOST = rac2_vip)(PORT = 1521))
            (ADDRESS = (PROTOCOL = TCP)(HOST = rac3_vip)(PORT = 1521))
            (LOAD_BALANCE = yes)
            (CONNECT_DATA =
              (SERVER = DEDICATED)
              (SERVICE_NAME = rac)
            )
  )
```

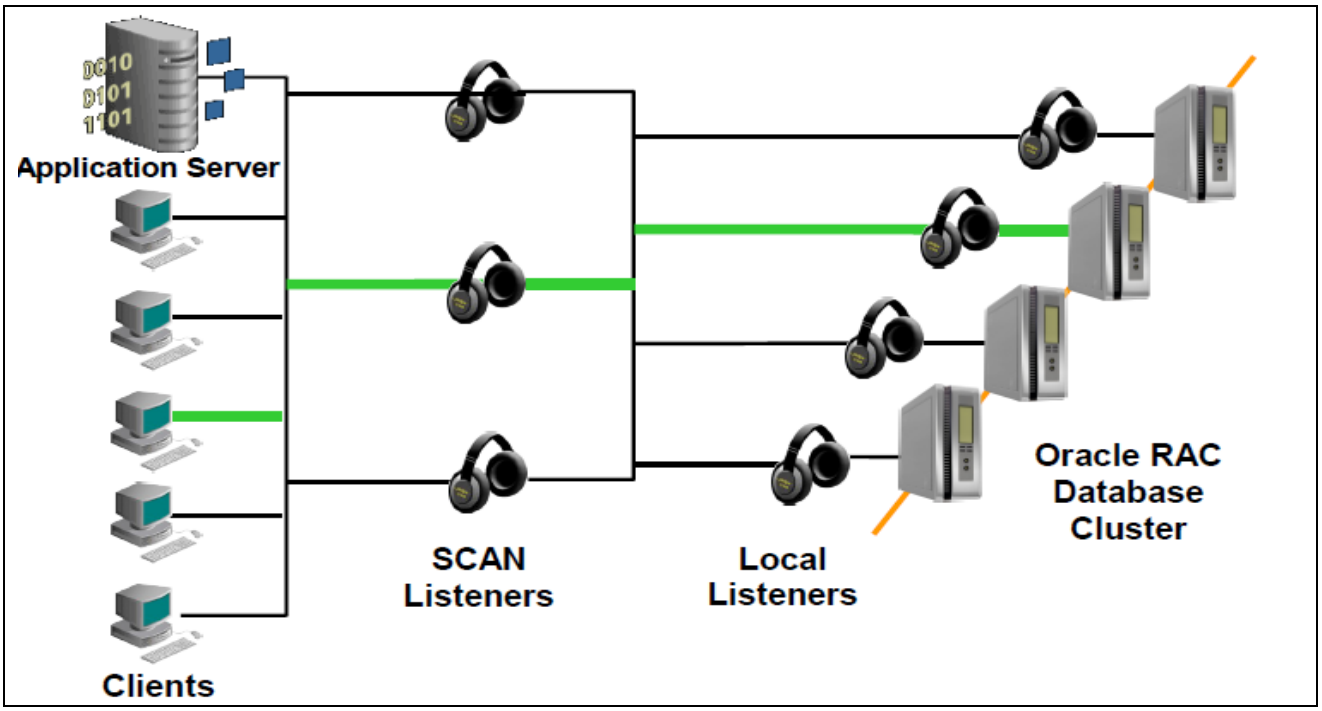
11gR2 버전의 tnsnames.ora

```
        RAC =
          (DESCRIPTION =
            (ADDRESS = (PROTOCOL = TCP)(HOST =cluster _scan_name)(PORT = 1521))
             (LOAD_BALANCE = yes)
            (CONNECT_DATA =
              (SERVER = DEDICATED)
              (SERVICE_NAME = rac)
            )
  )
```

```
EZconnet sqlplus system/manager@sales1-scan:1521/oltp
JDBC connect jdbc:oracle:thin:@sales1-scan:1521/oltp
```

SCAN 을 사용하지 않았던 예전 버전에서는 tnsnames 에 하나의 노드당 하나씩의 엔트리를 가지고 있으므로 노드의 변경이 있을 때에는 모든 클라이언트의 tnsnames.ora 를 수정해야 했으나, SCAN 을 쓰면 노드의 변경에 상관없이 하나의 엔트리의 정의 만으로 지속적으로 사용이 가능하다.

<구성도>



2.4.2. SCAN의 구성

DNS(Domain Name Service) 에 정의

DNS 에 정의하여 SCAN 을 사용할 때에는 하나의 도메인을 생성하여 3 개의 IP 로 연결한다. IP 가 3 개인 이유는 클러스터 내의 서버 대수와는 상관없고 로드발랜싱과 High-availability 를 위한 권장 값이다. IP 주소는 public network 와 같은 서브넷을 사용 하여야 하고 오라클 클러스터에서 관리하기 때문에 네트웍 인터페이스에 할당이 되면 안된다.

<sample> nslookup 명령을 이용하여 확인 가능하다.

```
sales1-scan.example.com    IN A 133.22.67.194
                           IN A 133.22.67.193
                           IN A 133.22.67.192
```

SCAN IP 는 라운드-로빈 알고리즘으로 Random 하게 look up 된다

<첫번째 nslookup>

```
[oracle@mynode] nslookup sales1-scan
Server: 131.32.249.41
Address: 131.32.249.41#53
Non-authoritative answer:
Name: sales1-scan.example.com
Address: 133.22.67.192
Name: sales1-scan.example.com
Address: 133.22.67.193
Name: sales1-scan.example.com
Address: 133.22.67.194
```

<두번째 nslookup>

```
[oracle@mynode] nslookup sales1-scan
Server: 131.32.249.41
Address: 131.32.249.41#53
Non-authoritative answer:
Name: sales1-scan.example.com
Address: 133.22.67.193
Name: sales1-scan.example.com
Address: 133.22.67.194
Name: sales1-scan.example.com
Address: 133.22.67.192
```

SCAN IP 로 접속 시 이 3 개의 IP 중 하나를 DNS 로부터 할당되고 이 SCAN 이 부하가 적은 Local listener 의 정보를 클라이언트에게 전달하게 되고 클라이언트는 할당 받은 local listener 로 접속을 맺게 된다.

GNS(Grid Naming Service)를 사용하여 SCAN 구성 시

GNS 를 사용할 때는 클러스터 구성시 DHCP 로부터 IP 를 자동으로 얻어온다.

DNS 와 GNS 를 사용하지 않고 SCAN 을 구성 시

SCAN 은 Oracle Grid Ingrastructure 설치 시 필수 구성 요소이므로 적당한 값이 입력되지 않으면 설치 시에 진행이 되지 않는다. DNS 나 GNS 구성이 안되어 있을 때 해결 법은 /etc/hosts 파일에 세개의 IP 가 아닌 하나의 IP 를 기입하여 해결할 수 있다. 그 대신 클러스터에 1 개의

SCAN 만이 첫번째 클러스터에 리소스로 등록이 된다. 전 버전에서 업그레이드 시에도 이런 방법으로 업그레이드 할 수 있다.

REOMTE_LISTENER 파라미터는 기본값이 SCAN listener 로 설정된다.

```
NAME                          TYPE        VALUE
----------------------------- ----------- ------------------------------
local_listener                string      (DESCRIPTION=(ADDRESS_LIST=(AD
                                          DRESS=(PROTOCOL=TCP)(HOST=133.
                                          22.67.111)(PORT=1521))))
remote_listener               string      sales1-scan.example.com:1521
```

2.5. Redundant Interconnect

Redundant interconnect 란 3rd-party 의 솔류션 없이 private network 를 다중화하는 기법으로 11gR2 부터 Grid Ingrastructure 에 의해 지원되고 HAIP 라고 한다.

HAIP 는 169.254.*.* 서브넷으로 자동으로 설정이 되고 다른 용도로는 사용 할 수 없으며 최대 4 개까지 생성 될 수 있다.

Private network 이 하나일 때 하나의 HAIP 가 추가되고 두 개일 때는 2 개의 HAIP 가 추가된다. 두 개 이상의 Private network 가 있을 때는 4 개의 HAIP 가 생성된다.

Private network 이 추가 되었을 때에는 모든 노드의 클러스터를 재시작 해야 새로운 HAIP 가 추가된다.

Single Private Network Adapter

```
$ $GRID_HOME/bin/oifcfg getif
eth1  10.1.0.128  global  cluster_interconnect
eth3  10.1.0.0  global  public

$GRID_HOME/bin/oifcfg iflist -p -n
eth1  10.1.0.128  PRIVATE  255.255.255.128
eth1  169.254.0.0  UNKNOWN  255.255.0.0
eth3  10.1.0.0  PRIVATE  255.255.255.128

ifconfig
eth1      Link encap:Ethernet  HWaddr 00:16:3E:11:11:22
          inet addr:10.1.0.168  Bcast:10.1.0.255  Mask:255.255.255.128
```

inet6 addr: fe80::216:3eff:fe11:1122/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:6369306 errors:0 dropped:0 overruns:0 frame:0
TX packets:4270790 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:3037449975 (2.8 GiB) TX bytes:2705797005 (2.5 GiB)

eth1:1 Link encap:Ethernet HWaddr 00:16:3E:11:22:22
inet addr:169.254.167.163 Bcast:169.254.255.255 Mask:255.255.0.0
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

Instance alert.log (ASM and database):

Private Interface 'eth1:1' configured from GPnP for use as a private interconnect.
[name='eth1:1', type=1, ip=169.254.167.163, mac=00-16-3e-11-11-22, net=169.254.0.0/16, mask=255.255.0.0, use=haip:cluster_interconnect/62]
Public Interface 'eth3' configured from GPnP for use as a public interface.
[name='eth3', type=1, ip=10.1.0.68, mac=00-16-3e-11-11-44, net=10.1.0.0/25, mask=255.255.255.128, use=public/1]
..
Shared memory segment for instance monitoring created
Picked latch-free SCN scheme 3
..
Cluster communication is configured to use the following interface(s) for this instance
169.254.167.163

Note: interconnect will use virtual private IP 169.254.167.163 instead of real private IP. For pre-11.2.0.2 instance, by default it will still use the real private IP; to take advantage of the new feature, init.ora parameter cluster_interconnects can be updated each time Grid is restarted .

```
SQL> select name,ip_address from v$cluster_interconnects;

NAME            IP_ADDRESS
--------------- ----------------
eth1:1          169.254.167.163
```

Multiple Private Network Adapters

```
$ $GRID_HOME/bin/oifcfg getif
eth1  10.1.0.128  global  cluster_interconnect
eth3  10.1.0.0  global  public
eth6  10.11.0.128  global  cluster_interconnect
eth7  10.12.0.128  global  cluster_interconnect

$ $GRID_HOME/bin/oifcfg iflist -p -n
eth1  10.1.0.128  PRIVATE  255.255.255.128
eth1  169.254.0.0  UNKNOWN  255.255.192.0
eth1  169.254.192.0  UNKNOWN  255.255.192.0
eth3  10.1.0.0  PRIVATE  255.255.255.128
eth6  10.11.0.128  PRIVATE  255.255.255.128
eth6  169.254.64.0  UNKNOWN  255.255.192.0
eth7  10.12.0.128  PRIVATE  255.255.255.128
eth7  169.254.128.0  UNKNOWN  255.255.192.0
```

Note: 4 개의 haip 가 생성 되었다. eth1 에는 2 개가 생성되었다.

```
ifconfig
eth1      Link encap:Ethernet  HWaddr 00:16:3E:11:11:22
          inet addr:10.1.0.168  Bcast:10.1.0.255  Mask:255.255.255.128
          inet6 addr: fe80::216:3eff:fe11:1122/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:15176906 errors:0 dropped:0 overruns:0 frame:0
          TX packets:10239298 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
```

```
          RX bytes:7929246238 (7.3 GiB)  TX bytes:5768511630 (5.3 GiB)

eth1:1    Link encap:Ethernet  HWaddr 00:16:3E:11:11:22
          inet addr:169.254.30.98  Bcast:169.254.63.255  Mask:255.255.192.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1

eth1:2    Link encap:Ethernet  HWaddr 00:16:3E:11:11:22
          inet addr:169.254.244.103  Bcast:169.254.255.255  Mask:255.255.192.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1

eth6      Link encap:Ethernet  HWaddr 00:16:3E:11:11:77
          inet addr:10.11.0.188  Bcast:10.11.0.255  Mask:255.255.255.128
          inet6 addr: fe80::216:3eff:fe11:1177/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:7068185 errors:0 dropped:0 overruns:0 frame:0
          TX packets:595746 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2692567483 (2.5 GiB)  TX bytes:382357191 (364.6 MiB)

eth6:1    Link encap:Ethernet  HWaddr 00:16:3E:11:11:77
          inet addr:169.254.112.250  Bcast:169.254.127.255  Mask:255.255.192.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1

eth7      Link encap:Ethernet  HWaddr 00:16:3E:11:11:88
          inet addr:10.12.0.208  Bcast:10.12.0.255  Mask:255.255.255.128
          inet6 addr: fe80::216:3eff:fe11:1188/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:6435829 errors:0 dropped:0 overruns:0 frame:0
          TX packets:314780 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2024577502 (1.8 GiB)  TX bytes:172461585 (164.4 MiB)

eth7:1    Link encap:Ethernet  HWaddr 00:16:3E:11:11:88
          inet addr:169.254.178.237  Bcast:169.254.191.255  Mask:255.255.192.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
```

Instance alert.log (ASM and database):

Private Interface 'eth1:1' configured from GPnP for use as a private interconnect.
[name='eth1:1', type=1, ip=169.254.30.98, mac=00-16-3e-11-11-22,
net=169.254.0.0/18, mask=255.255.192.0, use=haip:cluster_interconnect/62]
Private Interface 'eth6:1' configured from GPnP for use as a private interconnect.
[name='eth6:1', type=1, ip=169.254.112.250, mac=00-16-3e-11-11-77,
net=169.254.64.0/18, mask=255.255.192.0, use=haip:cluster_interconnect/62]
Private Interface 'eth7:1' configured from GPnP for use as a private interconnect.
[name='eth7:1', type=1, ip=169.254.178.237, mac=00-16-3e-11-11-88,
net=169.254.128.0/18, mask=255.255.192.0, use=haip:cluster_interconnect/62]
Private Interface 'eth1:2' configured from GPnP for use as a private interconnect.
[name='eth1:2', type=1, ip=169.254.244.103, mac=00-16-3e-11-11-22,
net=169.254.192.0/18, mask=255.255.192.0, use=haip:cluster_interconnect/62]
Public Interface 'eth3' configured from GPnP for use as a public interface.
[name='eth3', type=1, ip=10.1.0.68, mac=00-16-3e-11-11-44, net=10.1.0.0/25,
mask=255.255.255.128, use=public/1]
Picked latch-free SCN scheme 3

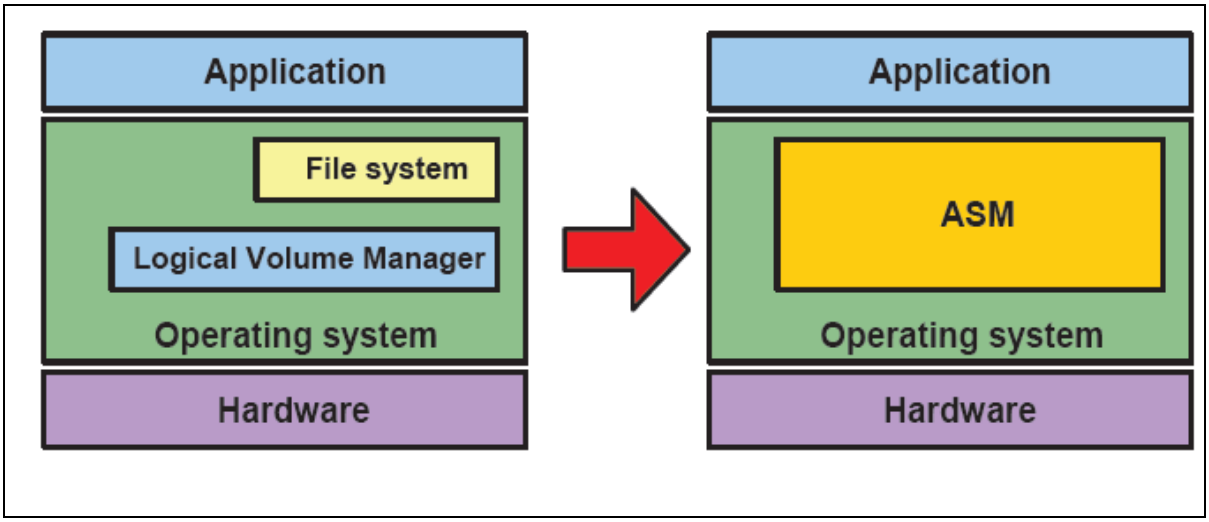
..
Cluster communication is configured to use the following interface(s) for this instance
169.254.30.98
169.254.112.250
169.254.178.237
169.254.244.103

Note:인터커넥터는 위의 4 개의 vip 를 사용하여 통신을 할 것이다. 만약 네트웍 장애가 나더라도 하나의 private network adapter 가 살아있는 한 4 개의 IP 들은 모두 active 상태로 남아있는다.

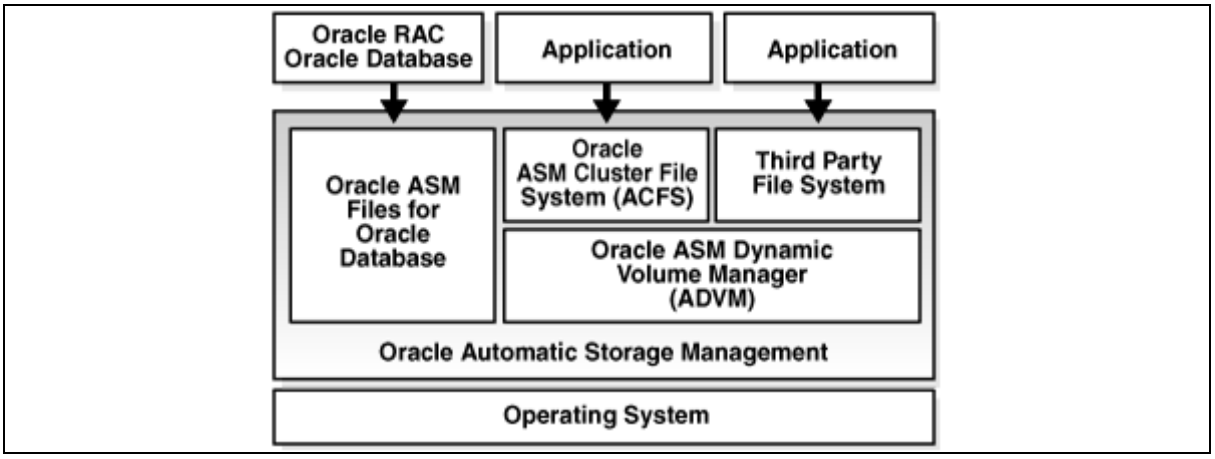
3. ASM Architecture

3.1. ASM(Automatic Storage Management)

ASM (Automatic Storage Management)은 볼륨 메니저와 파일 시스템이 오라클 데이터 베이스 서버로 구성된 것이다. 오라클에서 권장하는 최고의 오라클 데이터베이스 스토리지라 할 수 있다.



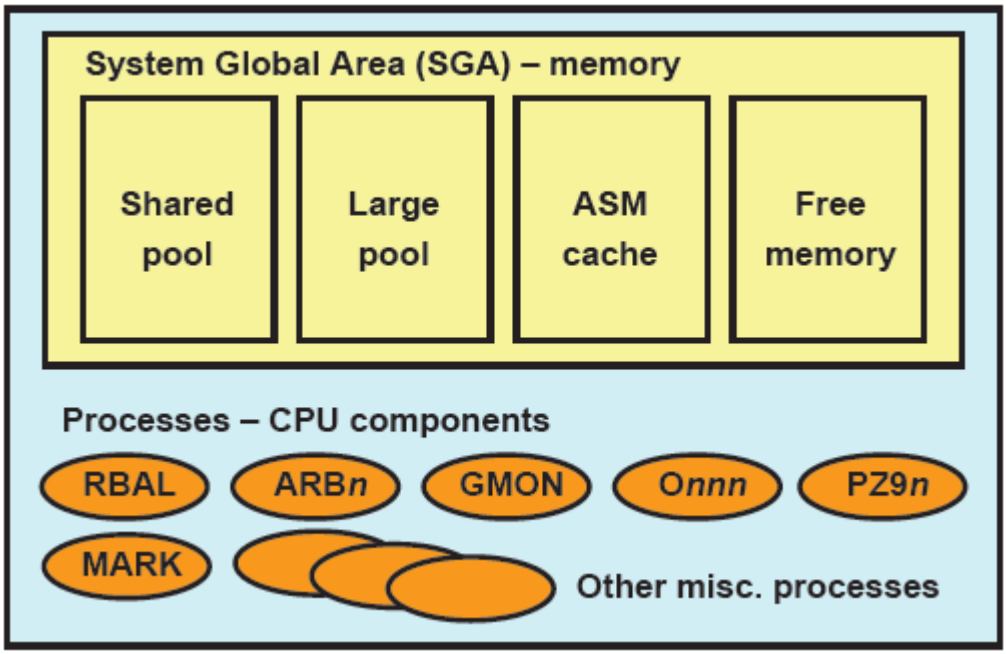
ASM Layer



Oracle ACFS (Oracle Automatic Storage Management Cluster File System)
ACFS 는 오라클 이외의 파일시스템을 관리하는 기술이며. ASM 과 통신을 할 수 있다. 오라클 홈 파일 시스템으로 사용 될 수 있으나 데이터 파일은 ACFS 에서 지원 하지 않는다. 데이터 파일은 ASM 에 구성을 해야한다.

3.2. ASM Components

ASM 도 SGA 영역과 백그라운드 프로세스 영역으로 구성 된다.데이터 베이스의 SGA 와는 약간 다른데 각각의 메모리와 프로세스에 대해 알아본다.



Memory

Memory	Description
Shared Pool	메타데이터 정보에 사용된다.
Large Pool	병렬 처리에 사용된다.
ASM Cache	리발랜싱 처리 시 블록을 읽고 쓰는데 사용된다.
Free Memory	할당되지 않은 free 메모리

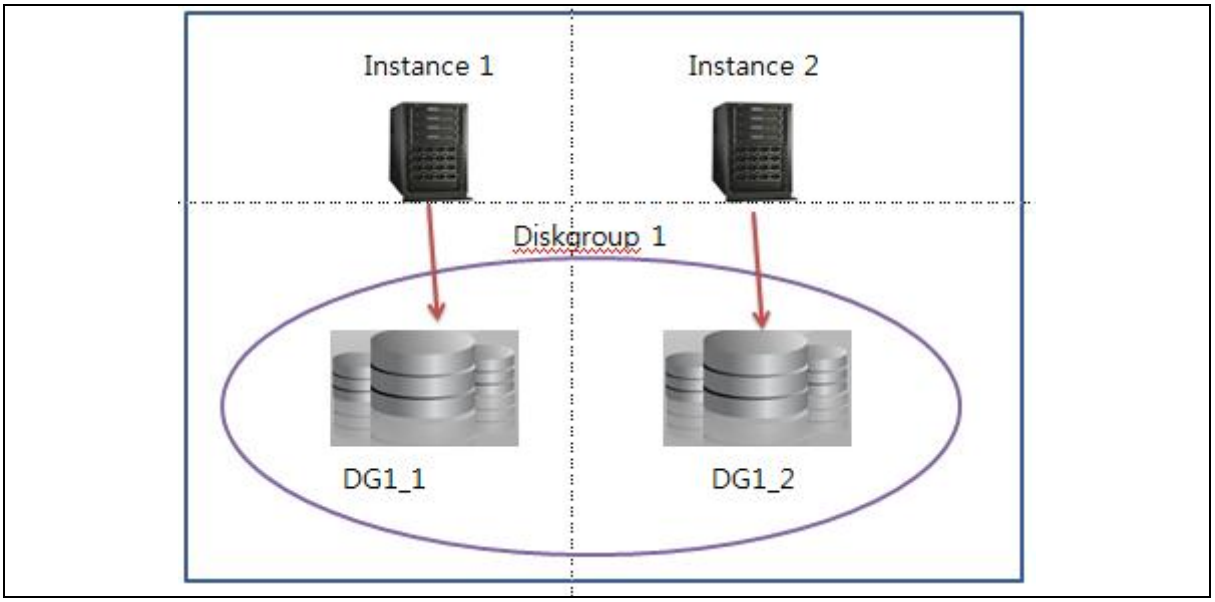
Processes

Processes	Description	Processes	Description
ARCn	The archiver processes	PSP0	The process spawner process
CKPT	The checkpoint process	QMNn	The queue monitor processes
DBWn	The database writer processes	RECO	The recoverer process
DIAG	The diagnosability process	SMON	The system monitor process
Jnnn	Job queue processes	VKTM	The virtual keeper of time process
LGWR	The log writer process	MMAN	The memory manager process
PMON	The process monitor process		

3.3. ASM 11g New Feature

3.3.1. Preferred Mirror Read

DG1_1 과 DG1_2 두개의 페일그룹을 갖는 Diskgroup 1 의 디스크 그룹이 있다고 하면 이전 버전에서는 DG1_1 에 원본을 ,DG1_2 에 사본을 기록하여 미러링을 하고 읽기를 수행 할 때 DG1_1 으로만 읽기가 수행 되었다. DG1_2 은 DG1_1 의 패일그룹이 접근이 불가능 할 상태에만 읽기가 수행 되었다.



11g 부터는 특정 노드가 원하는 특정 페일그룹으로부터 데이터를 읽어오도록 설정할 수 있다.클러스터의 노드들이 자신의 로컬디스크를 읽게 함으로서 성능을 향상 시킬수 있다.

```
INSTANCE 1
        SQL> alter system set asm_preferred_read_failure_groups =
        'DG1.DG1_1','DG2.DG1_1'
INSTANCE 2
        SQL> alter system set asm_preferred_read_failure_groups =
        'DG1.DG1_2','DG2.DG1_2'
```

위와 같이 설정 함으로 instance 1 은 DG1_1 로부터 읽기를 수행하고 DG1_1 접근이 불가능 할 때 DG1_2 에서 읽기를 수행한다.

Instance 2 는 DG1_2 로부터 읽기를 수행하고, DG1_2 접근이 불가능할 때 DG2_2 으로부터 읽기를 수행 한다.

3.3.2. ASM Fast Mirror Resync

Asm fast mirror resync 는 디스크 failure 시 restore 를 더욱 빠르게 하는 11g 의 새로운 기능이다. 10g 의 ASM 에서는 디스크 failure 시 디스크가 오프라인이 되고 drop 을 한뒤, 여분의 디스크에 resync 를 하였다. 이는 매우 비효율 적이었는데 11g 부터는 디스크 장애가 나게 되면 오프라인이 되어도 DISK_REPAIR_TIME 시간 까지 drop 되지 않는다. DISK_REPAIR_TIME (기본값 : 3.6h)시간 내에 디스크 복구되면 asm 은 모든 extents 를 resync 하는 것이 아니라 변경된 extents 만 resync 하게 된다. 이렇게 하므로 리스토어 시간을 줄일 수 있게 되었다. 이 기능을 사용하기 위해서는 DBMS 와 ASM 의 compatible 이 11.1 이상으로 설정 되어 있어야 한다.

<끝>