ASM Internals

By
Riyaj Shamsudeen
Me

- 20+ years as DBA
- OakTable member
- Oracle ACE Director Alumni
- Specializes in RAC, performance tuning and Internals.
- Slowly in to BigData

- rshamsud@orainternals.com
- orainternals.wordpress.com
- Web: www.orainternals.com
WARNING

Most of the topics in this presentations are from my research.

Writing about internals have issues:
   a. I completely misunderstood the data and trace files.
   b. Future version changed the feature, so, information is outdated.

Tested in version 11g, 12.1.0.2, Linux and Solaris 11 platform.
AGENDA

ASM overview: Instance, asmb etc
Tools: kfod, kfed, amdu
Disk group, redundancy, AU
ASM rebalance
Asmcmd
Conclusion
ASM is an Oracle Instance with instance_type=‘ASM’

ASM manages disks, luns and externalizes files to RDBMS

ASM instance is never opened. Simply in a mount state.

Demo: v$instance
Is ASM involved in I/O calls issued from RDBMS?
ASM provides an extent map of files to RDBMS.

RDBMS directly accesses the disk to perform I/O. ASM is not involved in I/O operation.

Extending files or adding data files will involve refresh of extent map from ASM to RDBMS.

Demo: v$instance
Architecture: With ASM
Truss of DBWR: ASM is not involved for RDBMS I/O to the devices.

Write calls to file pointer 262: (truss output)

/1: kaio(AIOWRITE, 262, 0x6DD3C000, 8192, 0xFC17E6380F4E4000) = 0
...
/1: kaio(AIOWRITE, 262, 0x7DF3F000, 49152, 0xFC17D7080BD8A000) = 0

File pointer 262 is a SCSI device (pfiles output)

   O_RDWR|O_NONBLOCK|O_DSYNC|O_LARGEFILE FD_CLOEXEC

/devices/iscsi/disk@0000iqm.demo.volumes-san0001,1:b,raw
ASM extent pointer array

$vsgastat$ shows the extent pointer array in the RDBMS. This array is retrieved from ASM instance.

```
select * from gv$s gastat where name like '%ASM extent%';
```

<table>
<thead>
<tr>
<th>INST_ID</th>
<th>POOL</th>
<th>NAME</th>
<th>BYTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>shared pool</td>
<td>ASM extent pointer array</td>
<td>171824</td>
</tr>
<tr>
<td>2</td>
<td>shared pool</td>
<td>ASM extent pointer array</td>
<td>171824</td>
</tr>
</tbody>
</table>

For large databases, this area will be bigger.

To improve instance startup performance, only minimal extent mapping is retrieved initially. More data added to this array on need basis.
How does RDBMS instance “knows” if local ASM instance fails?
RDBMS is a client (aka umbilical process)

- **asmb process running in RDBMS instance makes a connection to ASM instance**, as a foreground process for ASM instance.

- **asmb process sleeps in a loop and is the primary mechanism to detect ASM crash**.

- **If ASM instance crashes, asmb connection will die leading to an RDBMS instance crash.**

**Demo**: `asm_connections.sql`, `asm_clients.sql`
RDBMS as a client

Truss of a RDBMS startup shows that a LOCAL connection was made to the ASM instance.

1821: 2.9102 execve("/u02/app/11.2.0/grid/bin/oracle", 0x0E8E87F0, 0x0E9ED510)

Instance restart alone opens 6 different connections to ASM instance.

grep execve truss_startup.lst |grep grid
1821: 2.9102 0.0015 execve("/u02/app/11.2.0/grid/bin/oracle", 0x0E8E87F0, 0x0E9ED510) argc = 2
1941: 8.8772 0.0019 execve("/u02/app/11.2.0/grid/bin/oracle", 0x0E8E8090, 0x0EA11970) argc = 2
1966: 10.0884 0.0019 execve("/u02/app/11.2.0/grid/bin/oracle", 0x0E8E8090, 0x0EA11970) argc = 2
2002: 12.7198 0.0020 execve("/u02/app/11.2.0/grid/bin/oracle", 0x0E8E7550, 0x0E99B220) argc = 2
2010: 13.1669 0.0020 execve("/u02/app/11.2.0/grid/bin/oracle", 0x0E8E7550, 0x0E99B220) argc = 2
2066: 29.0296 0.0024 execve("/u02/app/11.2.0/grid/bin/oracle", 0x0E8E7550, 0x0E99B220) argc = 2
Death of asmb process

- asmb process sleeps on “ASM background timer” with 5s sleep cycle.

  *** 2011-10-05 00:38:11.486

  WAIT #0: nam='ASM background timer' ela= 5001967 p1=0 p2=0 p3=0 obj#=-1 tim=1247836548

- I killed the connection from ASM instance, resulting in asmb process death, followed by RDBMS instance crash

  NOTE: ASMB terminating

  Errors in file /u01/app/oracle/diag/rdbms/solrac/solrac1/trace/solrac1_asmb_1492.trc:
  ORA-15064: communication failure with ASM instance
  ORA-03113: end-of-file on communication channel

  Process ID:
  Session ID: 30 Serial number: 3

  ... ASMB (ospid: 1492): **terminating** the instance due to error 15064

Demo: Killing asmb connection
ASM disks

- During ASM startup, ASM instance scans the disks to identify all ASM disks.
- Parameter `asm_diskstring` identifies the disks to scan.
- To improve ASM startup time, set this parameter properly.
- For example,
  ```
  asm_diskstring = /dev/rdsk/c2t*d0s1
  ```
How to validate the asm_diskstring parameter from OS?
Kfod utility can be used to check all devices that qualifies asm_diskstring.

$ kfod status=TRUE asm_diskstring='/dev/mapper/' disks=ALL verbose=TRUE

<table>
<thead>
<tr>
<th>Disk</th>
<th>Size</th>
<th>Header</th>
<th>Path</th>
<th>User</th>
<th>Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>1:</td>
<td>20473 Mb MEMBER</td>
<td>/dev/mapper/asmdisk1p1</td>
<td>oracle oinstall</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2:</td>
<td>20473 Mb MEMBER</td>
<td>/dev/mapper/asmdisk2p1</td>
<td>oracle oinstall</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

ORACLE_SID ORACLE_HOME

+ASM1 /u01/app/12.1.0/grid
KFOD-00311: Error scanning device /dev/mapper/control
ORA-27041: unable to open file
Linux-x86_64 Error: 13: Permission denied
Additional information: 42
KFOD-00311: Error scanning device /dev/mapper/36000c29d5fb1e04764ebbedd94bb6acd
ORA-27041: unable to open file
Linux-x86_64 Error: 13: Permission denied

...
Must you have the same LUN path in all nodes of a RAC cluster?
**kfed disk header**

- kfed utility can be used to dump the metadata block(s) of the device.

- Without any parameter, kfed reads disk header.

$ kfed read /dev/rdsk/c2t9d0s1

```bash
kfbh.endian:       1 ; 0x000: 0x01

.. kfbh.type:       1 ; 0x002: KFBTYP_DISKHEAD

.. kfbh.block.blk:  0 ; 0x004: T=0 NUMB=0x0
kfbh.block.obj:    2147483655 ; 0x008: TYPE=0x8 NUMB=0x7...

kfdhdb.compat:     186646528 ; 0x020: 0x0b200000
kfdhdb.dsknum:     7 ; 0x024: 0x0007
kfdhdb.grptyp:     1 ; 0x026: KFDGTP_EXTERNAL
kfdhdb.hdrsts:     3 ; 0x027: KFDHDR_MEMBER
kfdhdb.dskname:    DATA_0007 ; 0x028: length=9
kfdhdb.grpname:    DATA ; 0x048: length=4
kfdhdb.fgname:     DATA_0007 ; 0x068: length=9

kfdhdb.capname:    ; 0x088: length=0
```

**Demo: kfed read**
<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>kfdhdb.secsize</td>
<td>512 ; 0x0b8: 0x0200</td>
</tr>
<tr>
<td>kfdhdb.blksize</td>
<td>4096 ; 0x0ba: 0x1000</td>
</tr>
<tr>
<td><strong>kfdhdb.ausize</strong></td>
<td><strong>1048576</strong> ; 0x0bc: 0x00100000</td>
</tr>
<tr>
<td>kfdhdb.mfact</td>
<td>113792 ; 0x0c0: 0x0001bc80</td>
</tr>
<tr>
<td>kfdhdb.dsksize</td>
<td>2000 ; 0x0c4: 0x000007d0</td>
</tr>
<tr>
<td>kfdhdb.pmcnt</td>
<td>2 ; 0x0c8: 0x00000002</td>
</tr>
<tr>
<td>kfdhdb.fstlocn</td>
<td>1 ; 0x0cc: 0x00000001</td>
</tr>
<tr>
<td>kfdhdb.altlocn</td>
<td>2 ; 0x0d0: 0x00000002</td>
</tr>
<tr>
<td>kfdhdb.flb1locn</td>
<td>0 ; 0x0d4: 0x00000000</td>
</tr>
<tr>
<td>kfdhdb.redomirrors[0]</td>
<td>0 ; 0x0d8: 0x0000</td>
</tr>
<tr>
<td>kfdhdb.redomirrors[1]</td>
<td>0 ; 0x0da: 0x0000</td>
</tr>
<tr>
<td>kfdhdb.redomirrors[2]</td>
<td>0 ; 0x0dc: 0x0000</td>
</tr>
<tr>
<td>kfdhdb.redomirrors[3]</td>
<td>0 ; 0x0de: 0x0000</td>
</tr>
<tr>
<td>kfdhdb.dbcompat</td>
<td>168820736 ; 0xe0: 0x0a100000</td>
</tr>
</tbody>
</table>

**Demo: kfed read**
ASM disks - RAC

揠 A LUN must be visible in all nodes of a cluster with proper permissions for ASM to consider a lun.

 yayg This means that the path need not be the same, but lun should exist and visible through asm_diskstring parameter.

揗 For example, same device have different names in two nodes:

    node1 /dev/rdsk/c2t9d0s1
    node2 /dev/rdsk/c2t11d0s1

揗 ASM identifies Lun even if configuration changes later

揗 Metadata kept in every disk header.

Demo: show parameter asm_diskstring
How to handle minor corruptions?
**kfed other blocks**

*kfed can be used to read other blocks in the lun also.*

```bash
$ kfed read /dev/rdsk/c2t9d0s1 aun=0 blkn=1 | grep kfbh.type
kfbh.type: 2 ; 0x002: KFBTYP_FREEESP

$ kfed read /dev/rdsk/c2t9d0s1 aun=0 blkn=2 | grep kfbh.type
kfbh.type: 3 ; 0x002: KFBTYP_ALLOC_TBL

# ASM also stores backup disk header in the second allocation unit, last 2 blocks.
$ kfed read /dev/rdsk/c2t9d0s1 aun=1 blkn=254 | more
kfbh.type: 1 ; 0x002: KFBTYP_DISKHEAD
kfbh.datfmt: 1 ; 0x003: 0x01
```

Demo: kfed read
Corrupting header

_minor header related repair possible_

$ kfed read /dev/mapper/asmdisk4p1 |more
kfbh.endian: 1 ; 0x000: 0x01
kfbh.hard: 130 ; 0x001: 0x82
kfbh.type: 1 ; 0x002: KFBTYP_DISKHEAD

$ dd if=/dev/zero of=/dev/mapper/asmdisk4p1 bs=1M count=1
1+0 records in
1+0 records out

$ kfed read /dev/mapper/asmdisk4p1 |more
kfbh.endian: 0 ; 0x000: 0x00
kfbh.hard: 0 ; 0x001: 0x00
kfbh.type: 0 ; 0x002: KFBTYP_INVALID

Demo: kfed read
Kfed repair

$ kfed repair /dev/mapper/asmdisk4p1

$ kfed read /dev/mapper/asmdisk4p1 |more

kfbh.endian: 1 ; 0x000: 0x01
kfbh.hard: 130 ; 0x001: 0x82
kfbh.type: 1 ; 0x002: KFBTYP_DISKHEAD

Demo: kfed read
$ amdu -diskstring=/dev/mapper/asmdisk3p1
amdu_2017_01_14_07_36_15/

$ ls -lt amdu_2017_01_14_07_36_15/
total 4
-rw-r--r--. 1 oracle oinstall 1834 Jan 14 07:36 report.txt

$ more amdu_2017_01_14_07_36_15/report.txt
--amdu--
******************************* AMDU Settings
*******************************
ORACLE_HOME = /u01/app/12.1.0/grid
System name: Linux
Node name: rac1.localdomain
Release: 3.8.13-44.el6uek.x86_64
Version: #2 SMP Fri Aug 8 21:59:01 PDT 2014
Machine: x86_64
amdu run: 14-JAN-17 07:36:15
Endianess: 1
Disk Path: /dev/mapper/asmdisk3p1

Unique Disk ID:

Disk Label:

Physical Sector Size: 512 bytes

Disk Size: 2047 megabytes

Group Name: TEST

Disk Name: TEST_0000

Failure Group Name: TEST_0000

Disk Number: 0

Header Status: 3

Disk Creation Time: 2017/01/11 23:49:59.434000

Last Mount Time: 2017/01/14 07:32:17.969000

Compatibility Version: 0x0a100000(10010000)

Disk Sector Size: 512 bytes

Disk size in AUs: 2047 AUs

Group Redundancy: 2

Metadata Block Size: 4096 bytes

AU Size: 1048576 bytes

Stride: 113792 AUs
What is the critical difference between ASM mirroring and OS mirroring?
ASM disk group

- Picture of a Disk group with Normal redundancy. Two failure groups are allocated since this is a mirrored disk group.

- ASM does not mirror disks, rather extents are kept in two separate failure groups.
Example

Construct the failure groups such a way that one component failure affects at the most one failure group.

create diskgroup DATA normal redundancy

Failure group fl1 disk
‘/dev/rdsk/c3t11d3s4’, ‘/dev/rdsk/c3t11d4s4’, ‘/dev/rdsk/c3t11d5s4’,
‘/dev/rdsk/c3t11d6s4’

Failure group fl2 disk
‘/dev/rdsk/c4t12d3s4’, ‘/dev/rdsk/c4t12d4s4’, ‘/dev/rdsk/c4t12d5s4’,
‘/dev/rdsk/c4t11ds4’

Failure group fl3 disk
‘/dev/rdsk/c5t13d3s4’, ‘/dev/rdsk/c5t13d4s4’, ‘/dev/rdsk/c5t13d5s4’,
‘/dev/rdsk/c5t13ds4’

Failure group fl4 disk
‘/dev/rdsk/c6t14d3s4’, ‘/dev/rdsk/c6t14d4s4’, ‘/dev/rdsk/c6t14d5s4’,
‘/dev/rdsk/c6t14ds4’;
Hierarchy of ASM files, disks, etc
ASM files – Normal redundancy

ASM files are allocated from mirrored extents between the failure groups.

Demo: asm_file_analysis.sql
ASM files – Normal redundancy - Exadata

ASM files are allocated from mirrored extents between the failure groups.

Demo: asm_file_analysis.sql
Extents vs Files

- ASM files are allocated as series of extents.
- ASM extents are made up of one or more allocation units.

ASM extents are contained within an ASM disk though.
Extent vs AU

• 1 extent = 1 AU up to 20000 extents. 1 extent=8 AUs after 20000 extents.
• This is one asm file and so extents are distributed between the devices (striping).

• Source: Concepts manual
Allocation_unit (AU)

Allocation unit defines a smallest size disk segment that can be allocated, at disk group level.

Allocation_unit defaults to 1MB. It can be increased in multiples of 2 i.e. 2, 4, 8, 16MB etc while creating a diskgroup. (11g). 4MB recommended.

Increased allocation_unit is useful in VLDB daabases. 32MB/64MB possible if compatible>=11.1.

Once a disk group is created with an allocation unit it can not be altered.

In 10g, underscore parameters _asm_ausize can be used to modify the allocation_unit.
AU & Striping (course grained)

- Stripe size = 1MB for better read size such as data files
- AU size = 1 MB

Source: Concepts manual
AU & Striping (fine grained)

- Stripe size = 128KB for higher latency files such as control file and redo log files
- AU size = 1 MB

• Source: Concepts manual
Rebalance
Rebalance

Addition or deletion of asm disk from a disk group will trigger a rebalance operation.

Extents are moved from existing disks to new disks, rebalancing the disk usage.
Processing details

- RBAL is triggered when there is addition/deletion/resize of disks.
- RBAL acts as a co-ordinator process, updates metadata that ASM rebalance is underway.
- Determines the extent to move and the target disk. Hands off the work to ARBx process.
- ARBx process moves the extent and replies back to RBAL after the successful completion.
- This goes on until RBAL completes the rebalance operation.
THANK YOU

Email: rshamsud@orainternals.com
Blog: orainternals.wordpress.com
Web: www.orainternals.com