

ASM support utilities

Some examples and comments on the use of ASM support utilities, in particular **kfed** and **amdu**. These utilities allow troubleshooting of ASM diskgroups also with **unmounted diskgroups**. Beware, kfed in write mode can corrupt your ASM config too. Note: when ASM diskgroups can be mounted similar info are available in X\$ tables as shown in ASM_Internals. Oracle support note ID 553639.1 has useful info on amdu and kfed.

AMDU

- **Allows to dump ASM contents without opening diskgroups, allows to check ASM file mirroring when using normal redundancy**
- powerful tool for troubleshooting, introduced in 11g, usable in 10g too.
- amdu run creates a directory with a `report.tx` file, plus optionally the `.map` and `.img` dump files (for metadata and data, the actual output depends on the switches specified in the command line)
- a few examples of interest:

```
# displays online help for the utility
$ amdu -help

# extracts file 267 from ASM diskgroup TEST4_DATADG1
# Note: works as asmcmd cp but also on dismounted disk groups!
$ amdu -dis '/dev/mapper/itsto*p1' -extract TEST4_DATADG1.267

# compares primary and mirror extents in normal redundancy disk groups
# Useful to check for potential corruption issues
# results in the report.txt file
$ amdu -dis '/dev/mapper/itsto*p1' -compare -extract TEST4_DATADG1.267 -noextract

# dump contents of a given diskgroup and does not create image file
# the .map file reports on all files found (column number 5 prefixed by the letter F)
$ amdu -dis '/dev/mapper/itsto*p1' -noimage -dump TEST4_DATADG1

# print 10 blocks for the first extent of file 267
$ amdu -dis '/dev/mapper/itsto*p1' -print TEST4_DATADG1.F267.X1.B1.C10
```

Example: how to retrieve DB files from a dismounted diskgroup

- extract file 256 from the data diskgroup as it often the first copy of controlfile
- (or use a diskgroup dump as above to find files list. db alert log and RMAN catalog are other sources to check)
- use strings command to find the list of files to be retrieved and extract them with amdu
- `-fullscan`, `-baddisks` and/or `-former` can be of help for some cases

```
$ amdu -dis '/dev/mapper/itsto*p1' -extract TEST4_DATADG1.256
$ strings TEST4_DATADG1.256|grep TATEST4_DATADG1|less # find list of <FILENUM>
$ amdu -dis '/dev/mapper/itsto*p1' -extract TEST4_DATADG1.<FILENUM>
```

KFED

- kfed can be used to read and write ASM metadata. In particular disk headers and ASM (hidden) metadata files.
- note: kfed in write mode is a powerful but potentially dangerous tool

```
# displays online help for the utility
$ kfed -help

# reads the disk header to stdout
$ kfed op=read dev=/dev/mapper/itstor741_11p1
```

ASM_utilities < PDBService < TWiki

```
# reads the specified AU and block into file /tmp/a
$ kfed op=read dev=/dev/mapper/itstor741_11p1 aunum=3 blknum=3 text=/tmp/a

# writes from /tmp/a into the specified AU and block
#block checksum is computed and written together with data
$ kfed op=write dev=/dev/mapper/itstor741_11p1 aunum=3 blknum=3 text=/tmp/a
```

Example: how to change an ASM diskgroup parameter on a dismounted disk group

- the parameter is in ASM (hidden) file #9
- find AUs that contain that information (can be up to 3 mirrored copies in a normal redundancy diskgroup with 3 or more failgroups)
- read current value into a flat file, update the file and write the updated file with checksum using kfed
- dismount and mount diskgroup to see change (query v\$asm_attribute)

```
SQL> select DISK_KFFXP,AU_KFFXP,LXN_KFFXP,SIZE_KFFXP from x$kffxp where NUMBER_KFFXP=9 and GROUP
```

```
DISK_NUMBER PATH
```

```
-----
41 /dev/mapper/itstor741_11p1
22 /dev/mapper/itstor739_10p1
34 /dev/mapper/itstor740_9p1
```

```
READ: kfed read /dev/mapper/itstor741_11p1 aunum=3 blknum=3 text=/tmp/a
EDIT: vi /tmp/a and change attribute value (for example change smart_scan_enable from TRUE to FALSE)
to disable exadata smart scans in this particular diskgroup, note update length field to
```

Example:

```
'cell.smart_scan_capable'='FALSE';
kfede[0].name: smart_scan_capable ; 0x034: length=18
kfede[0].value: FALSE ; 0x074: length=5
kfede[0].length: 5 ; 0x174: 0x0005
```

```
'cell.smart_scan_capable'='TRUE';
kfede[0].name: smart_scan_capable ; 0x034: length=18
kfede[0].value: TRUE ; 0x074: length=4
kfede[0].length: 4 ; 0x174: 0x0004
```

```
WRITE PRIMARY EXTENT: kfed write /dev/mapper/itstor741_11p1 aunum=3 blknum=3 text=/tmp/a
WRITE SECONDARY extents:
kfed write /dev/mapper/itstor739_10p1 aunum=3 blknum=3 text=/tmp/a
kfed write /dev/mapper/itstor740_9p1 aunum=2 blknum=3 text=/tmp/a
```

```
SQL> (dismount) and mount the group again to see the change in v$asm_attribute
Note after changing the parameter you may need to explicitly turn smart scans off on the RDBMS
alter system set cell_offload_processing=FALSE scope=both sid='*';
```

Example: how to undrop ASM diskgroup and/or recover files from dropped diskgroups

- method 1: use amdu to recover the files: amdu -dis '/dev/mapper/devstor4_1p1' -former -extract RDTEST2_TESTDROP.256
- method 2: use kfed to change the disk headers back to member

```
kfed read /dev/mapper/devstor4_1p1 aunum=0 blknum=0 text=devstor4_1p1.txt

vi devstor4_1p1.txt and change
from:
kfdhdb.hdrsts: 4 ; 0x027: KFDHDR_FORMER
to:
kfdhdb.hdrsts: 3 ; 0x027: KFDHDR_MEMBER

kfed write /dev/mapper/devstor4_1p1 aunum=0 blknum=0 text=devstor4_1p1
```

Example: how to rename an ASM disk

- In the case of ASM 12c, the procedure is:
 - ◆ alter diskgroup .. mount restricted
 - ◆ alter diskgroup rename disk '...' to 'new path';
- This is an undocumented and unsupported procedure that can be used in 11g (at your own risk):

```
1) Identify all the mirror extents of ASM file N.2 for the relevant diskgroup (i.e. the disk)
select disk_kffxp, au_kffxp, path from x$kffxp x, v$asm_disk_stat v
where x.group_kffxp=2 and number_kffxp=2
      and x.group_kffxp=v.group_number and x.disk_kffxp=v.disk_number
order by x.pxn_kffxp;
```

```
2) dismount the diskgroup
```

```
3) Read the first copy of the disk header and update it with the new disk name and length.
```

```
kfed read /dev/mapper/disklname_p1 aunum=0 blknum=0 text=disklname_p1_au0blk0
```

```
make a backup copy of the file disklname_p1_au0blk0
```

```
vi disklname_1p1_au0blk0
```

```
update -> kfddde[0].dskname:                NEWNAME ; 0x038: length=7
```

```
kfed write /dev/mapper/disklname_p1 aunum=0 blknum=0 text=disklname_1p1_au0blk0
```

```
Note: it does not seem to be necessary to update the other copies of the disk header, ASM
```

```
4) Use kfed to read, modify and then write the new disk name in the relevant entry of the disk header.
```

```
This is similar to point N.3 with the notable minor differences that:
```

```
- From step (1) above we know in which aunum and disk we will find the directory but not the file.
```

```
- For diskgroups in normal or high redundancy it's advisable to update all the mirror copies.
```

```
5) mount the diskgroup, monitor the ASM alert log and hope for the best :)
6) Some tips to recover from errors:
```

```
- dismount the diskgroup
```

```
- copy back the previous values of the modified blocks with using from the backup files
```

```
- mount the diskgroup
```

```
- the affected disks will probably be offline, online them back.
```

How to use KFED and AMDU in 10g

kfed and amdu are available by default in 11g in \$ORACLE_HOME/bin. In 10g one needs

- For kfed a link operation needs to be run:

```
cd $ORACLE_HOME/rdbms/lib
make -f ins_rdbms.mk kfed
```

- for amdu, one needs to download it from support. See note note ID 553639.1
- in alternative an installation of and additional ORACLE_HOME with 11gR2 binaries should also work for the purposes of using amdu

BBED

BBED is an Oracle internal utility to do BLOCKEDIT. It is extensively discussed at

http://www.orafaq.com/papers/dissassembling_the_data_block.pdf BBED doesn't work on ASM by default.

A work-around has been studied by <http://oracleprof.blogspot.com/2009/06/asm-and-bbed.html> An example to get started can be found here below.

Note that bbed does not build out of the box on 11g and 12c, as there are a few files missing in the Oracle binaries distribution to properly build bbed. There is a simple workaround, that consists on copying the relevant files from 10g binaries before running make. More details at this blog entry by Laurent Leturgez

<http://laurent-leturgez.com/2011/06/29/bbed-in-oracle-11g/>

```
cd $ORACLE_HOME/rdbms/lib
make -f ins_rdbms.mk $ORACLE_HOME/rdbms/lib/bbed
```

ASM_utilities < PDBService < TWiki

Dump an extent or an ASM file with the methods described above, for example:
amdu -dis '/dev/mapper/itsto*p1' -extract TEST4_DATADG1.555

Edit: vi bbed.par (or rather follow the example in the ASMBBED link above)
blocksize=8192
datafile=/ORA/dbs01/oracle/home/work/amdu_2010_02_01_17_22_39/TEST2_DATADG1_555.f
mode=browse

Finally:
\$ORACLE_HOME/rdbms/lib/bbed parfile=bbed.par

Revisions:

First version, Jan 2010, Luca.Canali@cernNOSPAMPLEASE.ch

Additional examples for amdu and kfed, May 2014, Luca.Canali@cernNOSPAMPLEASE.ch

This topic: PDBService > ASM_utilities

Topic revision: r15 - 2015-06-23 - LucaCanali



Copyright &© 2008-2022 by the contributing authors. All material on this collaboration platform is the property of the contributing authors.

or Ideas, requests, problems regarding TWiki? use Discourse or Send feedback