ACFS und DBFS im Vergleich

Daniel Hillinger
Value Transformation Services
München

Schlüsselworte
DBFS, ACFS, Filesystem

Einleitung
Der Vortrag gibt einen Überblick über die beiden Filesysteme von Oracle im Datenbankumfeld, ASM Cluster Filesystem (ACFS) und Database Filesystem (DBFS) und ihre Einsatzmöglichkeiten.
Folgende Fragestellungen werden behandelt:

- Wie kann man ACFS und DBFS implementieren?
- Was darf man nicht tun?
- Welche Replikationsmöglichkeiten gibt es für Desaster Recovery?
- Welche Probleme sind bei der Verwendung von ACFS bzw. DBFS unter Umständen zu erwarten?
- Welche Performance kann man erwarten?

In diesem Skript werden nur der Aufbau eines ACFS und DBFS beschrieben.

Installation ACFS
Voraussetzung ist eine laufende Oracle Clusterware (Grid Infrastructure) und eine Diskgruppe. In meinem Beispiel heißt die Diskgruppe DATA und der Eigentümer der Grid Infrastructure Installation wird im Folgenden als Benutzer grid abgekürzt.

```
grid$ asmcmd volcreate -G DATA -s 20G MY_ACFS
```

Hiermit wird das logische Volume angelegt, es ist ab jetzt als Block device unter /dev/asm verfügbar.

```
grid$ ls /dev/asm/my_acfs*
my_acfs_472
root# mkfs -t acfs /dev/asm/my_acfs_472
root# mkdir /acfs
root# srvctl add filesystem -device /dev/asm/my_acfs_472 -volume my_acfs -
diskgroup data -mountpoint /acfs –user grid
root# srvctl start filesystem –device /dev/asm/my_acfs_472
root# chown grid:dba /acfs
grid$ df -Th /acfs
Filesystem           Type   Size  Used Avail Use% Mounted on
/dev/asm/my_acfs_472   acfs   100G   44G   57G  44% /acfs
grid$ crsctl stat res ora.data.my_acfs.acfs -t
--------------------------------------------------------------------------------
NAME           TARGET  STATE        SERVER                   STATE_DETAILS
--------------------------------------------------------------------------------
Local Resources
--------------------------------------------------------------------------------
ora.data.my_acfs.acfs
               ONLINE  ONLINE       host1              mounted on /acfs
               ONLINE  ONLINE       host2              mounted on /acfs
```

Mit den letzten beiden Befehlen wird überprüft, ob alles ordnungsmäßig funktioniert hat. Der Mountpoint muss auf allen Clusternodes angelegt werden.

Damit das Filesystem auch für Clients zu Verfügung steht, muss es per NFS exportiert werden. Mit Oracle Clusterware 12c gibt es eine neue Funktion, High Available Network Filesystem (HANFS), die ein einfaches Einbinden einer NFS Ressource im Cluster möglich macht. Bei einer OCW 11g muss man das Ganze zu Fuß machen, mit einer vip und einem eigenen Script. Hier beschreibe ich die Einrichtung für HANFS. Oracle unterstützt für HANFS nur NFS Version 3 (über IPv4) und kein Locking. Die Dienste, portmap und nfs, müssen gestartet sein und außerdem müssen sie beim Systemstart mit hochfahren. Die folgenden Befehle konfigurieren und starten die HAVIP und den Export:

```
root# srvctl add havip -id ACFS1 -address ACFS1.domain.com -netnum 1 -
description „MY ACFS“
root# srvctl add exportfs -path /acfs -id ACFS1 -option „rw,no_root_squash“
-clients client.domain.com
root# srvctl start exportfs -id ACFS1
```

Der Status kann mit betriebssystemeigenen Kommandos wie auch mit Clustermitteln geprüft werden, z.B.: exportfs -v, crsctl stat res
Das Einbinden eines NFS Shares ist nur noch eine Kleinigkeit. Im Vorhinein kann man prüfen, ob die Freigabe sichtbar ist, dies geht mit showmount -e acfs1.domain.com . Danach einfach folgende Zeile in der /etc/fstab hinzufügen und mit mount einhängen:

```
acfs1.domain.com:/acfs /acfs nfs
rw,bg,hard,nointr,nolock,rsize=32768,wsize=32768,tcp,actimeo=0,vers=3,timeo
=600 0 0
root# mount /acfs
```

Installation DBFS
Voraussetzung für den Aufbau eines DBFS ist eine laufende Oracle Clusterware (Grid Infrastructure) und eine Datenbank. Der Eigentümer der Datenbankinstallation wird im Folgenden als Benutzer oracle abgekürzt.
Ein eigener Service für das Filesystem ist empfehlenswert, um das Filesystem alleine stoppen zu können. Dies ist besonders für die Hochverfügbarkeitstests sinnvoll.

```
oracle$ srvctl add service -d dbfsdb -s dbfsservice –g pool1 –e select –m
basic –w 5 –z 180
```

Bei einer RAC Datenbank ist es wichtig den Service mit entsprechender failover method und failover type zu erstellen.

```
sql> select failover_method, failover_type from dba_services where
service_name = 'dbfsservice';
BASIC      SELECT
```

Die Dateien werden intern als Large Object (LOB) abgelegt, dafür gibt es einen Parameter der beeinflusst, ob LOBs als BasicFile oder als SecureFile angelegt werden:

DB_SECUREFILE

Property	Description
Parameter type	String
Syntax	`DB_SECUREFILE = { NEVER | PERMITTED | ALWAYS | IGNORE }`
Default value	`PERMITTED`
Modifiable	`ALTER SESSION`, `ALTER SYSTEM`
Basic	No

Abb. 1 Auszug aus der Oracle Dokumentation Version 11.2

Für DBFS ist SercureFiles die bessere Wahl, d.h. der Parameter muss auf Always stehen. Verpflichtend sind ein eigener Tablespace und User, ebenso wir die Rechte create session, create table, create view, create procedure, dbfs_role.

```
sql> create tablespace tbdbfs
     datafile '+DATA' size 10g autoextend on next 1g
     EXTENT MANAGEMENT LOCAL AUTOALLOCATE  SEGMENT SPACE MANAGEMENT AUTO ;
sql> create user dbfs
     identified by dbf5DBF5
     default tablespace tbdbfs quota unlimited on tbdbfs;
sql> grant create session
     ,create table,create view,create procedure
     ,dbfs_role
     to dbfs;
```

Das Filesystem wird durch das SQL Script $ORACLE_HOME/rdbms/admin/dbfs_create_filesystem.sql angelegt. Dieses Script braucht 2 Übergabeparameter, den Tablespace und den Name des Filesystems.

```
sql> connect dbfs/dbf5DBF5
Connected.
sql> @?/rdbms/admin/dbfs_create_filesystem.sql tbdbfs my_dbfs
```

Das Script liefert ziemlich viel Output. Diesen darf sich jeder bei seinem eigenen Aufbau ansehen ;)
Jetzt hat man zwar ein Filesystem, aber es ist Unix-seitig nicht zu erreichen. Der wesentlich aufwendigere Teil ist das Einbinden als unix-style Filesystem.
Diese zeige ich anhand eines rehl 6.
Da der DBFS mount ein Filesystem in Userspace (FUSE) Implementierung ist, müssen folgende RPMs vorhanden sein. Version kann sich unterscheiden.

- fuse-2.8.3-4.0.2.el6.x86_64
- fuse-devel-2.8.3-4.0.2.el6.x86_64
- fuse-libs-2.8.3-4.0.2.el6.x86_64

Nun muss der Oracle Client installiert werden, mit der Option Administrator (oracle.install.client.installType=Administrator). Falls schon eine Oracle Database Server Installation vorhanden ist, kann auch diese genutzt werden. Wichtig ist nur, dass es in $ORACLE_HOME/bin ein Binary mit dem Namen dbfs_client gibt.

Ebenfalls wegen der FUSE-Implementierung muss der User oracle der Gruppe fuse hinzugefügt werden, damit er alle notwendigen Skripte Ausführen darf.

```
root# usermod -a -G fuse oracle
```

Damit das Filesystem von allen Benutzern genutzt werden kann, benötigt es einen Eintrag in /etc/fuse.conf. Diese Datei muss auch von der Gruppe fuse gelesen werden können.

```
root# echo user_allow_other > /etc/fuse.conf
root# chgrp fuse /etc/fuse.conf
root# chmod 644 /etc/fuse.conf
```

2 Bibliotheken werden benötigt und damit diese für alle verfügbar sind:

```
root# mkdir -p /usr/local/lib
root# cp $ORACLE_HOME/lib/libnnz11.so /usr/local/lib/
root# cp $ORACLE_HOME/lib/libclntsh.so.11.1 /usr/local/lib/
root# chown oracle:oinstall /usr/local/lib/libnnz11.so
/usr/local/lib/libclntsh.so.11.1
root# echo /usr/local/lib >> /etc/ld.so.conf.d/usr_local_lib.conf
root# ldconfig
```

Mit ldconfig -v kann überprüft werden, ob alles geklappt hat. Noch den mountpoint anlgegen:

```
root# mkdir /dbfs
```

Nun muss der Oracle Client noch konfiguriert werden. Dazu benötigt er eine tnsnames.ora und eine sqlnet.ora. Diese beiden Dateien können im standard Pfad $ORACLE_HOME/network/admin abgelegt werden oder man legt ein neues Verzeichnis an und setzt die Variable $TNS_ADMIN. Damit das Filesystem direkt beim Hochfahren gemountet werden kann, muss ein autologin wallet verwendet werden.

```
tnsnames.ora:
MY_DBFS =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = <IP>)(PORT = 1521))
    (CONNECT_DATA = (SERVER = DEDICATED) (SERVICE_NAME = dbfsservice))
  )
sqlnet.ora:
WALLET_LOCATION =
  (SOURCE=(METHOD=FILE)(METHOD_DATA=(DIRECTORY=<PFAD>))
  )
SQLNET.WALLET_OVERRIDE = TRUE
```

Zum Anlegen des Wallets und Eintragen der Credentials:

```
oracle$ mkdir -p $TNS_ADMIN/wallet
oracle$ mkstore -wrl $TNS_ADMIN/wallet –create
#Passwort eingeben
oracle$ mkstore -wrl $TNS_ADMIN/wallet -createCredential MY_DBFS dbfs
dbf5DBF5
```

Ab jetzt könnten wir das Filesystem verwenden, allerdings kann es bei Start noch nicht automatisch gemountet werden und es ist noch ziemlich unhandlich. Um das Ganze einfacher zu machen, werden

wir noch ein mount und umount Skript zur Verfügung stellen und einen Eintrag in der /etc/fstab machen.

```
root# cat /sbin/mount.dbfs
#!/bin/bash
export ORACLE_HOME=<PFAD>
nohup $ORACLE_HOME/bin/dbfs_client $@ &>/dev/null &
root# cat /sbin/umount.dbfs
#!/bin/bash
fusermount -u $@
root# chmod 750 /sbin/mount.dbfs /sbin/umount.dbfs
root# chgrp fuse /sbin/mount.dbfs /sbin/umount.dbfs
root# grep dbfs /etc/fstab
/sbin/mount.dbfs#/@MY_DBFS /dbfs fuse
rw,user,allow_other,noauto,direct_io,server_readahead=0,failover 0 0
root# mount /dbfs
```

Wichtig ist zu überprüfen, ob alle Benutzer das Filesystem sehen. Dies kann man einfach mit einem df testen.

```
root# df -Th /dbfs
dbfs-@MY_DBFS:/   fuse   7.2G  204M  7.0G   3% /dbfs
```

Tipp: In der Datei /proc/mounts steht der mount auch drin, wenn der für den aktuellen Benutzer nicht sichtbar ist.
Nun ist das DBFS eingerichtet und kann verwendet werden. Zu beachten ist noch, dass es unter dem Mountpoint ein Verzeichnis mit dem beim Anlegen spezifizierten Namen des DBFS gibt. Nur innerhalb dieses Verzeichnisses kann man die Dateien ablegen.

Kontaktadresse:
Daniel Hillinger
Value Transformation Services
Am Tucherpark 12
D-80538 München

Telefon:	+49 (0) 89 378 28737
E-Mail	daniel.hillinger@de.ibm.com
Internet:	www.v-tservices.com