Oracle Maximum
Availability Architecture

Oracle Database Appliance: Implementing Disaster Recovery Solutions Using Oracle Data Guard

Protect production systems while leveraging standby computing power

ORACLE WHITE PAPER | SEPTEMBER 2017

ORACLE

.

Introduction

Oracle Database Appliances are a pre-built, pre-tuned, and ready-to-use non-clustered and clustered database systems that includes servers, storage, networking, and software in an optimized configuration that makes them easy to deploy, operate, and manage. An Oracle Database Appliance is a complete and ideal database platform for small, medium, and large sized database implementations and incorporates robust, time-tested Oracle technologies, including the world leading Oracle Database, the best selling Oracle Real Application Clusters (RAC) database option, Oracle Clusterware, and Oracle Automatic Storage Management (ASM). By integrating hardware and software, Oracle Database Appliance eliminates the complexities inherent in non-integrated, manually assembled database solutions, reducing deployment time from weeks or months to just a few hours, while preventing configuration and setup errors that often result in sub-optimal, hard-to-manage database environments.

Data Protection using Oracle Active Data Guard

While the Oracle Database Appliance is a highly available system in itself, a standby database environment can provide protection against planned and unplanned downtime as well as protection against data loss in case the primary database environment becomes unavailable or corrupted. With Oracle Maximum Availability Architecture (MAA) practices, the standby database can be synchronized with the primary database, thereby providing minimal database downtime for planned maintenance activities such as database upgrades and for unplanned outages such as data corruptions, database failures, cluster failures or major disasters. A standby database has therefore always been an integral component of MAA to provide additional high availability and data protection for any mission critical production system.

Oracle Data Guard is the most comprehensive solution available to eliminate single points of failure for mission critical Oracle Databases. It prevents data loss and downtime in the simplest and most economical manner by maintaining a synchronized physical replica of a production database at a remote location. If the production database is unavailable for any reason, client connections can quickly, and in some configurations transparently, failover to the synchronized replica to restore service.

Oracle Active Data Guard enables administrators to improve performance by offloading processing from the primary database to a physical standby database that is open read-only while it applies updates received from the primary database. Offload capabilities of Oracle Active Data Guard 12c were enhanced to include read-only reporting and ad-hoc queries (including DML to global temporary tables and unique global or session sequences), data extracts, fast incremental backups, redo transport compression, efficient servicing of multiple remote destinations, and the ability to extend zero data loss protection to a remote standby database without impacting primary database performance. Oracle Active Data Guard also increases high availability by performing automatic block repair and enabling High Availability Upgrades (new automation in Oracle Database 12c for more easily implementing database rolling upgrades)

Oracle recommends using a separate, dedicated Oracle Database Appliance system to host the Data Guard standby system for a mission critical production system running on the primary Oracle Database Appliance system. If the standby database resides in the same data center or resides in another datacenter within the same metro region, applications can be configured to transparently fail over to the standby for any Data Guard role transition. This Data Guard configuration provides high availability, data protection and some local disaster protection. If the standby database resides across geographical regions, this Data Guard configuration typically provides additional disaster recovery protection.

Benefits of using Oracle Data Guard

Oracle Data Guard provides numerous benefits and enables greater efficiency and efficacy for the deployed architecture. With the use of Oracle Active Data Guard, the standby database environment does not need to be idle, dark capacity. Instead, the standby database can actively serve many useful purposes. These additional uses greatly increase the overall return on effort and investment.

Migration to Oracle Database Appliance - If you plan to migrate existing databases to Oracle Database Appliance, then Oracle Data Guard enables an easy approach for migration of your databases to Oracle Database Appliance. You can simply setup a Physical Standby database on your Oracle Database Appliance and switchover operations from the legacy environment to the new Oracle Database Appliance environment. This includes migration across certain platforms as well. For example, to migrate your databases currently running on the Windows platform to Oracle Database Appliance, a Linux platform, you may simply setup Oracle Data Guard between the two environments and perform a switchover. This approach to platform migration provides the flexibility to switchback, if for any reason you choose to do so after testing. Refer to My Oracle Support (MOS) note 413484.1 Data Guard Support for Heterogeneous Primary and Physical Standbys in Same Data Guard Configuration, for more information about platform migration using Oracle Data Guard.

Note: Oracle Data Guard also allows you to migrate across database versions using a transient logical standby database.

Disaster Recovery - Oracle Data Guard physical standby database provides an ideal solution for disaster protection. Disaster scenarios vary from burst water or steam pipes, fire, hurricanes, vandalism, to earthquakes, floods, and acts of terrorism. Oracle Data Guard Physical Standby Database maintains a block-for-block copy of the production database. In the event the primary environment becomes unavailable due to any reason, the standby environment can be quickly activated to maintain continued database availability for your applications.

High Availability – The standby database can also be useful in maintaining availability during planned and unplanned outages and downtimes. Such events may include configuration changes, hardware replacements, and so forth as well as data corruptions, failures resulting from human errors, and other unexpected system component or complete system failures.

Database Rolling Upgrades – The standby database minimizes downtime when patch bundles are applied and changes are made to the primary Oracle Database Appliance. Patches or other maintenance is applied first at the standby database, validated, and then production workload is switched from the primary to the standby system. The only downtime for the databases is the short period of time required to change roles between primary and standby. Please refer to My Oracle Support (MOS) note 1265700.1, Oracle Patch Assurance - Data Guard Standby-First Patch Apply, for more information.

Offloading Workload and Activities – Despite its name, the standby environment does not have to be idle. It can be actively used to maximize the overall return on your investment. With a physical standby database in place, several key activities can be offloaded to the standby environment. These include:

- **Read-Only Workload** – Using Oracle Active Data Guard option, the standby database can be open for read only query workload while being in the standby mode and accepting redo log updates from the primary database. In many cases, offloading read only workload to the standby database can dramatically reduce the production workload, thereby increasing the overall available capacity for the production system.
- **Backups** – Because the Oracle Data Guard physical standby database is a block-for-block copy of the primary database, database backups can be completely offloaded to the standby environment and these backups can be transparently used to restore and recover the primary database in the event of a failure or database loss. Note that if Oracle Active Data Guard option is licensed, then fast incremental backups can be run at the standby database, further adding to the appeal of offloading backups to the standby database.
- **Block Repair** – One of the other benefits of the physical standby database is the ability that it provides to automatically recover from block corruption scenarios. In a primary/standby configuration a corrupt block can be automatically repaired and this operation can be

completely transparent to the end user and database administrator. The Block Repair feature is also a part of the Oracle Active Data Guard option.

» **Snapshot Standby** – The Snapshot Standby database is an updatable standby database that provides full data protection for the primary database. It continues to receive redo data from the primary but the apply process is halted while the standby database is open for read/write operations for testing purposes. When testing is complete, a single command reverts the standby database back to its original state, discarding the changes made while it was open in read-write mode and applying the accumulated redo logs to make it synchronize with the current state of primary database.

Configuration Best Practices

This section describes some of the important best practices for setting up Oracle Data Guard on Oracle Database Appliance. For a complete list of general Oracle Data Guard best practices, which also apply to the Oracle Database Appliance environment, please refer to Oracle Maximum Availability Architecture and Oracle Data Guard best practices available at http://www.oracle.com/technetwork/database/features/availability/oracle-database-maa-best-practices-155386.html

» **Match the primary and standby database configuration** -- In order to maintain consistent service levels and to use the primary and standby databases transparently, it is important to match the resources, setup, and configuration of the primary and standby systems as much as possible. Significant differences between the primary and standby database configuration can result in sub-optimal performance and unpredictable behavior when role transitions occur. Specifically, the following recommendations should be considered:

» ***Run Primary and Standby Database on Separate Oracle Database Appliances*** - It is recommended that the primary and the standby databases run on separate, dedicated Oracle Database Appliance units preferably located in a geographically distant location.

» ***Run Primary and Standby Database in Same Configuration*** -- Three different database configurations are supported on Oracle Database Appliance; Oracle RAC database, Oracle RAC One, and Single-Instance Enterprise Edition database. The standby database should also be of the same configuration type as the primary database. Thus, if the primary database is configured as an Oracle RAC database, then the standby database should also be configured as an Oracle RAC database.

» ***Size Primary and Standby Instances Similar to Each Other*** -- The instances on the primary and standby databases should be configured similar to each other in terms of database parameter settings including memory, CPU, networking, and storage. This helps avoid any unpredictability when the database switch roles. In addition, any operating system configuration customizations should be mirrored in the two environments.

» **Configure Flashback Database on both Primary and Standby Databases** -- The Flashback Database feature enables rapid role transitions and reduces the effort required to re-establish database roles after a transition. As a best practice, Flashback Database should be configured on both the primary and the standby database. If FLASHBACK is only deemed necessary by you for the purposes of re-instantiation, then it would be a good practice to reduce the retention time from the default 24 hours to 2 hours.

» **Use Dedicated Network for Standby Traffic** -- Oracle Database Appliance comes pre-built with multiple redundant network interfaces. If required, a separate network path can be configured for the standby traffic to minimize any performance impact on the user and application related workload. Note that since Oracle Data Guard needs to transport only the changes made to the primary database from the primary database to the standby database, it does not impose any unnecessary requirements on the network than is needed. Therefore, many deployments of Oracle Data Guard may not require a separate network path for redo log transport between primary and standby. However, some high volume applications or your organizations best practices and standards may require a separate network path for redo log transport. Oracle Database Appliance does provide additional network interfaces on each server node that can be used for this purpose. Please refer to MOS note 1422563.1 for additional details on configuring a dedicated network for disaster recovery purposes on Oracle Database Appliance.

» **Utilizing Oracle Active Data Guard** – Oracle Active Data Guard allows for read only standby of near current data since redo apply remain continuously active between primary and standby environments. This can help distribute or offload the read-only workload from the primary environment to the standby database, increasing the return on investment in the standby database. Note that with Oracle Active Data Guard, fast incremental backups can be run on the standby database. The fast incremental backups could potentially reduce backup windows from hours to minutes. Additionally, Active Data Guard with real time apply enables auto-block corruption repair.

» **Review Oracle Maximum Availability Architecture (MAA) Best practices for Oracle Database -** Depending on your deployment and usage of the Data Guard environment and other requirements, you may find many MAA Best Practices such as the following useful.

- Client Failover Best Practices for Data Guard 12c
- Best Practices for Configuring Redo Transport for Active Data Guard 12c
- Best Practices for Asynchronous Redo Transport - Data Guard and Active Data Guard
- Best Practices for Synchronous Redo Transport - Data Guard and Active Data Guard
- Best Practices for Automatic Resolution of Outages to Resume Data Guard Zero Data Loss
- Role Transition Best Practices: Data Guard and Active Data Guard
- Preventing, Detecting, and Repairing Block Corruption - Oracle Database 12c

Please refer to http://www.oracle.com/technetwork/database/features/availability/oracle-database-maa-best-practices-155386.html for the above best practices and more.

Oracle Data Guard Setup Between Oracle Database Appliance Systems

Oracle Data Guard Configuration Procedures

Depending on the version of the primary database, different methods can be used for setting up the Data Guard Physical Standby Database environment.

Database Versions 11.2 - The standard RMAN DUPLICATE method is recommended for database versions 11.2. Although, this method also works for higher versions, there are other options available which are described below. Refer to Creating a Physical Standby Database using RMAN Duplicate (RAC or Non-RAC) MOS Note document 1617946.1 for details.

> Note: An example step-by-step procedure for creating a primary-standby configuration for Oracle 11g databases using Oracle Database Appliance platforms is provided in Appendix B of this white paper.

Database Versions 12.1 and 12.2 - You can also use the RMAN 'restore… from service ' method if the database version is 12.1.0.2 or higher. Refer to Creating a Physical Standby database using RMAN restore... from service MOS Note document 2283978.1 for details on how to instantiate the standby database using the 'restore… from service' method. The RMAN 'restore... from service' clause enables online restore and recover of primary database files to a standby database over a network. This method also allows for utilizing the SECTION SIZE clause for parallelization of the restore over multiple RMAN channels.

> Note: An example step-by-step procedure for creating a primary-standby configuration for Oracle 12c databases using Oracle Database Appliance platforms is provided in Appendix A of this white paper.

As you follow the above documents for setting up your primary and standby database environments in an Oracle Data Guard configuration, adhere to the following guidelines that are specific to the Oracle Database Appliance platform.

1. You may not currently use Oracle Enterprise Manager for instantiating a standby system and using Oracle Database Appliance as the platform. This is due to a bug. You can however, follow the above mentioned notes or examples provided in the appendix sections of this white paper for configuring your 11g and 12c environments.
2. If using Oracle ACFS storage, pre-create database storage on the standby Oracle Database Appliance system prior to standby database instantiation. Use the "oakcli create dbstorage" command as the root user to create ACFS storage for your standby database before you instantiate the standby database. For example,

   # oakcli create dbstorage -db stbydb

   Please refer to the example step-by-step configuration procedures listed in the appendix section of this white paper.
3. You may use the standby database deployment procedures on Oracle Database Appliance Bare Metal as well as Oracle Database Appliance Virtualized Platform deployments.

Oracle Database Appliance Bare Metal and Virtualized Platform Configurations

Oracle Database Appliance can be configured as a Bare Metal (non-virtualized) platform or as a Virtualized Platform. The Oracle Data Guard Physical Standby setup process outlined in this white paper can be used in both Oracle Database Appliance configurations, i.e., Bare Metal and Virtualized Platform. On Oracle Database Appliance Virtualized Platform, the configuration steps are executed within the ODA_BASE domain. In addition, Virtual LANs can be used on Oracle Database Appliance Virtualized Platform for configuring a logically separate network for disaster recovery purposes.

Oracle Database Appliance Small, Medium, and Large Platform Configurations

Oracle Real Application Clusters (RAC) and Oracle Data Guard are fundamental and essential components of Oracle Maximum Availability Architecture (MAA). While you can also setup Oracle Data Guard configuration between Oracle Database Appliance X6-2 S|M|L hardware models (the smaller, single node configurations), because these systems are single node systems, such configurations do not adhere to MAA guidelines as you can only run Oracle Real Application Clusters (RAC) on Oracle Database Appliance X6-2 HA hardware model.

Conclusion

Oracle Data Guard enables you to instantly deploy an effective disaster recovery protection strategy right from the time of initial deployment of your Oracle Database Appliance. You can use the Oracle Data Guard Physical Standby environment for multiple purposes besides as a disaster recovery solution. The physical standby configuration and setup process outlined in this white paper is quick, simple, and it can be completed without any downtime incurred on the primary database. Most of the standby creation steps are automated using tools such as Oracle Appliance Manager, Database Configuration Assistance (DBCA), RMAN, and Oracle Data Guard and can be used for setting up standby databases on Oracle Database Appliance.

Appendix A 12c Example setup on Oracle Database Appliance

Example Environment

The following section describes the primary and standby database environment topologies used in the subsequent Data Guard setup example using Oracle Database Appliance.

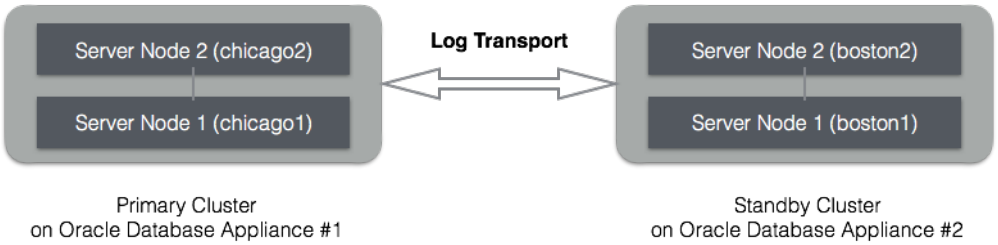

Figure 1 Configuration Topology of Oracle RAC on Oracle Database Appliance

	Primary Oracle Database Appliance		**Standby Oracle Database Appliance**	
Appliance Name	appliance#1		appliance#2	
Host Names	proddb1	proddb1	stbydb1	stbydb2
Cluster Name	PCLUSTER		SCLUSTER	
Database Name	chicago		chicago	
Database Unique Name	chicago		boston	
Instance Name	chicago1	chicago2	boston1	boston2
SCAN Name and IPs	proddb-scan (10.1.27.2, 10.1.27.3)		stbydb-scan (10.1.27.4, 10.1.27.5)	
Grid Infrastructure Software Installation	/u01/app/12.1.0.2/grid		/u01/app/12.1.0.2/grid	
Oracle Database Software Installation	/u01/app/oracle/product/12.1.0.2/db_home1		/u01/app/oracle/product/12.1.0.2/db_home1	
ARCHIVELOG mode	Yes		Yes	
FORCE LOGGING mode	Yes		Yes	

Table 1 - Example Oracle Database Primary and Standby Configuration

Primary Environment Configuration

1. Create Standby Redo Logs

Standby Redo Logs (SRLs) host redo data received from the primary database. In advance of the primary standby setup, Oracle recommends that standby redo logs be created on the primary database as well so that it is immediately ready to receive redo data following a switch-over to the standby role.

Create Standby Redo Logs (SRL) on the primary database. Each thread of the standby redo log must have at least one more redo log group than the corresponding thread of the online redo log. For example,

```
SQL> alter database add standby logfile thread 1 group 7 size 1G, group 8 size 1G, group 9 size 1G,
group 10 size 1G;
SQL> alter database add standby logfile thread 2 group 11 size 1G,group 12 size 1G, group 13 size 1G,
group 14 size 1G;
```

To check the number of online redo logs & their sizes, use the following query.

```
SQL> select thread#, group#, bytes/1024/1024/1024 SIZE_IN_GB, status from v$log;
```

Note that the size of the standby redo logs should match the size of the redo logs. On the Oracle Database Appliance platform, the standby redo logs have to be created on the REDO disk group which resides on the solid state disks.

To validate the size of each log file and number of log groups in the standby redo log, use the following query.

```
SQL> select group#, thread#, bytes from v$standby_log;
```

2. Enable archivelog mode on primary database

Archiving is the process of saving and protecting REDO information in the form of archive files before the redo logs of an active database are overwritten in a circular manner. Database created on Oracle Database Appliance have archiving turned on by default. However, it is not mandatory to run your databases in archive log mode.

Verify that the primary database is running in ARCHIVELOG mode.

```
SQL> archive log list
```

If the primary database is not running in ARCHIVELOG mode, then enable ARCHIVELOG mode as follows.

Shutdown both instances on Oracle Database Appliance.

```
$ srvctl stop database –d proddb
```

Startup mount one instance in exclusive mode.

```
SQL> startup mount exclusive;
```

Turn on archiving.

```
SQL> alter database archivelog;
```

Shutdown the instance.

```
SQL> shutdown immediate;
```

Restart the database.

```
$ srvctl start database –d proddb
```

3. Enable FORCE LOGGING mode.

Force logging enables you to capture database operations performed with the NOLOGGING attribute. This ensures integrity of your standby database. Verify if FORCE LOGGING is already enabled on your primary database.

```
SQL> select force_logging from v$database;
```

If FORCE LOGGING is not enabled, then enable it using the following commands.

```
SQL> alter database force logging;
```

4. Configure Flashback Database feature

The Oracle Flashback Database feature provide a fast alternative to performing incomplete database recovery. Although using the Flashback Database feature is optional, it can be very useful for faster re-instatement of the old primary database after a failover. Thus, if you do a failover to the standby, and the old primary can be repaired, you do not have to rebuild the old primary database as a standby database but simply flashback and let Oracle Data Guard resynchronize from that point onwards.

Check if the primary database has Flashback Database enabled, and if required enable it.

```
SQL> select flashback_on from v$database;
SQL> alter database flashback on;
```

Note that enabling Flashback Database will require additional space consumption in the Fast Recovery Area (RECO Disk Group). The space used by flashback logs can be controlled by setting the parameter DB_FLASHBACK_RETENTION_TARGET to a desired value. This value is specified in minutes. For example,

```
SQL> alter system set DB_FLASHBACK_RETENTION_TARGET=120 scope=both sid='*';
```

5. Enable Standby File Management

When the primary database adds or drops a datafile, the corresponding action should also be automatically taken on the standby database. This operation can be enabled using automated standby file management.

```
SQL> alter system set STANDBY_FILE_MANAGEMENT=AUTO scope=both sid='*';
```

6. Setup TNS Entries and Listeners

Oracle Net Service Names must be configured to enable redo transportation across the databases. Update tnsnames.ora file to include the TNS alias for both primary and standby databases. Note that in the Oracle Database Appliance, the tnsnames.ora file is located in network/admin directory of the Oracle database home.

```
$ vi $ORACLE_HOME/network/admin/tnsnames.ora
```

Primary

```
chicago =
 (DESCRIPTION =
  (ADDRESS = (PROTOCOL = TCP)(HOST = proddb-scan)(PORT = 1521))
  (CONNECT_DATA =
   (SERVER = DEDICATED)
   (SERVICE_NAME = chicago)
  )
 )

boston =
 (DESCRIPTION =
  (ADDRESS = (PROTOCOL = TCP)(HOST = stbydb-scan)(PORT = 1521))
  (CONNECT_DATA =
```

```
    (SERVER = DEDICATED)
    (SERVICE_NAME = boston)
  )
 )
```

on proddb1

```
chicago_local_listener =
 (DESCRIPTION =
  (ADDRESS = (PROTOCOL = TCP)(HOST = proddb1-vip.us.oracle.com)(PORT = 1521)))
```

on proddb2

```
chicago_local_listener =
 (DESCRIPTION =
  (ADDRESS = (PROTOCOL = TCP)(HOST = proddb2-vip.us.oracle.com)(PORT = 1521)))
```

Standby

```
chicago =
 (DESCRIPTION =
  (ADDRESS = (PROTOCOL = TCP)(HOST = proddb-scan)(PORT = 1521))
  (CONNECT_DATA =
   (SERVER = DEDICATED)
   (SERVICE_NAME = chicago)
  )
 )

boston =
 (DESCRIPTION =
  (ADDRESS = (PROTOCOL = TCP)(HOST = stbydb-scan)(PORT = 1521))
  (CONNECT_DATA =
   (SERVER = DEDICATED)
   (SERVICE_NAME = boston)
  )
 )
```

on stbydb1

```
boston_local_listener =
 (DESCRIPTION =
  (ADDRESS = (PROTOCOL = TCP)(HOST = stbydb1-vip.us.oracle.com)(PORT = 1521)))
```

on stbydb2

```
boston_local_listener =
 (DESCRIPTION =
  (ADDRESS = (PROTOCOL = TCP)(HOST = stbydb2-vip.us.oracle.com)(PORT = 1521)))
```

7. Create a pfile from the spfile on the primary database.

```
[oracle@proddb1]$ export ORACLE_HOME=/u01/app/oracle/product/12.1.0.2/dbhome_1
[oracle@proddb1]$ export ORACLE_SID=chicago1
[oracle@proddb1]$ export PATH=$ORACLE_HOME/bin:$PATH
[oracle@proddb1]$ sqlplus / as sysdba
SQL> create pfile='/tmp/chicago.pfile' from spfile;
```

8. Add/modify the parameters on the Primary/Standby. For example:

Parameters to be modified on the Standby as compared to the Primary	
Primary	**Standby**
***.cluster_database=TRUE** **chicago2.instance_number=2** **chicago1.instance_number=1** **chicago2.thread=2** **chicago1.thread=1** **chicago2.undo_tablespace='UNDOTBS2'** **chicago1.undo_tablespace='UNDOTBS1'** **…….** **…….** ***.db_unique_name=chicago** ***.remote_listener='proddb-scan:1521'** ***.local_listener='chicago_local_listener'** ***.DB_FILE_NAME_CONVERT='boston','chicago','BOSTON' ,' CHICAGO'** ***.LOG_FILE_NAME_CONVERT='boston','chicago','BOSTON','CHICAGO'** ***.LOG_ARCHIVE_DEST_1='LOCATION=USE_DB_RECOVERY_FILE_DEST VALID_FOR=(ALL_LOGFILES,ALL_ROLES) DB_UNIQUE_NAME=chicago'** **<u>add the following parameters</u>** ***.audit_file_dest=/u01/app/oracle/admin/chicago/adump** ***.fal_server='boston'** ***.remote_login_passwordfile='exclusive'** **<u>add only on X5-2 and X6-2 HA</u>** ***._cluster_flash_cache_slave_file=''** ***.db_flash_cache_file="/u02/app/oracle/oradata/flashdata/.ACFS/snaps/flashcache/chicago/flash1"**	***.cluster_database=TRUE** **boston2.instance_number=2** **boston1.instance_number=1** **boston2.thread=2** **boston1.thread=1** **boston2.undo_tablespace='UNDOTBS2'** **boston1.undo_tablespace='UNDOTBS1'** **…….** **…….** ***.db_unique_name=boston** ***.remote_listener='stbydb-scan:1521'** ***.local_listener='boston_local_listener'** ***.DB_FILE_NAME_CONVERT='chicago','boston','CHICAGO','BOSTON'** ***.LOG_FILE_NAME_CONVERT='chicago','boston','CHICAGO','BOSTON'** ***.LOG_ARCHIVE_DEST_1='LOCATION=USE_DB_RECOVERY_FILE_DEST VALID_FOR=(ALL_LOGFILES,ALL_ROLES) DB_UNIQUE_NAME=boston'** **<u>add the following parameters</u>** ***.audit_file_dest=/u01/app/oracle/admin/boston/adump** ***.fal_server='chicago'** ***.remote_login_passwordfile='exclusive'** **<u>add only on X5-2 and X6-2 HA</u>** ***._cluster_flash_cache_slave_file=''** ***.db_flash_cache_file="/u02/app/oracle/oradata/flashdata/.ACFS/snaps/flashcache/boston/flash1"**

9. On all standby hosts create the audit directory for the boston database.

[oracle@stbydb1] mkdir -p /u01/app/oracle/admin/boston/adump

[oracle@stbydb2] mkdir -p /u01/app/oracle/admin/boston/adump

10. Create the filesystem structure on the standby

```
[root@stbydb1]# oakcli create dbstorage -db boston
INFO: 2017-08-12 06:16:44: Please check the logfile
/opt/oracle/oak/log/stbydb1/tools/12.1.2.10.0/createdbstorage_boston_69182.log' for more details
...
SUCCESS: All nodes in /opt/oracle/oak/onecmd/tmp/db_nodes are pingable and alive.
INFO: 2017-08-14 04:47:45: Successfully setup the storage structure for the database 'boston'
INFO: 2017-08-14 04:47:45: Set the following directory structure for the Database boston
INFO: 2017-08-14 04:47:45: DATA: /u02/app/oracle/oradata/datastore/.ACFS/snaps/boston
INFO: 2017-08-14 04:47:45: REDO: /u01/app/oracle/oradata/datastore/boston
INFO: 2017-08-14 04:47:45: RECO: /u01/app/oracle/fast_recovery_area/datastore/boston
SUCCESS: 2017-08-14 04:47:45: Successfully setup the Storage for the Database : boston
```

11. Password Copy

Copy the password file from the primary database to the first standby host.

```
[oracle@proddb1]$ srvctl config database -d chicago |grep Password
Password file: /u02/app/oracle/oradata/datastore/.ACFS/snaps/chicago/chicago/orapwchicago

[oracle@proddb1]$ scp /u02/app/oracle/oradata/datastore/.ACFS/snaps/chicago/chicago/orapwchicago
oracle@stbydb1.us.oracle.com:/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston/orapwboston
```

12. Copy the modified pfile to the first standby host and mount the standby database.

Make a note of the path where is the standby control file is created.

```
[oracle@proddb1]$ scp /tmp/chicago.pfile oracle@stbydb1.us.oracle.com:/tmp/boston.pfile
[oracle@stbydb1]$ export ORACLE_HOME=/u01/app/oracle/product/12.2.0.1/dbhome_1
[oracle@stbydb1]$ export ORACLE_SID=boston1
[oracle@stbydb1]$ export PATH=$ORACLE_HOME/bin:$PATH
[oracle@stbydb1]$ cp /u01/app/oracle/product/12.1.0.2/dbhome_1/dbs/orapwboston
/u01/app/oracle/product/12.1.0.2/dbhome_1/dbs/orapwboston1
[oracle@stbydb1]$ rman target /
RMAN> startup nomount pfile='/tmp/boston.pfile';
RMAN> restore standby controlfile from service chicago;
Starting restore at 12-AUG-17
using target database control file instead of recovery catalog
allocated channel: ORA_DISK_1
channel ORA_DISK_1: SID=162 instance=boston1 device type=DISK
channel ORA_DISK_1: starting datafile backup set restore
channel ORA_DISK_1: using network backup set from service Chicago
channel ORA_DISK_1: restoring control file
channel ORA_DISK_1: restore complete, elapsed time: 00:00:26
output file name=/u01/app/oracle/oradata/datastore/boston/BOSTON/controlfile/o1_mf_drw8zb81_.ctl
Finished restore at 12-AUG-17
```

13. Update the Control File parameter

Edit the pfile '/tmp/chicago.pfile' and replace the control_files parameter to show the new path from the previous output. For example: control_files= '/u01/app/oracle/oradata/datastore/boston/BOSTON/controlfile/o1_mf_drw8zb81_.ctl'

```
[oracle@stbydb1]$ vi /tmp/boston.pfile
```

14. Start the Standby instance

Start the standby instance in nomount mode using the modified pfile, create the spfile and restart the instance with the spfile.

```
[oracle@stbydb1$ export ORACLE_HOME=/u01/app/oracle/product/12.2.0.1/dbhome_1
[oracle@stbydb1$ export ORACLE_SID=boston1
[oracle@stbydb1$ export PATH=$ORACLE_HOME/bin:$PATH
[oracle@stbydb1$ mkdir -p /u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston
[oracle@stbydb1]$ sqlplus / as sysdba
SQL> startup nomount force pfile='/tmp/boston.pfile';
SQL> create spfile='/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston/spfileboston.ora' from
pfile='/tmp/boston.pfile';
SQL> !echo "spfile='/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston/spfileboston.ora'" >
/u01/app/oracle/product/12.1.0.2/dbhome_1/dbs/initboston1.ora
SQL> startup mount force;
```

15. Enable Parallelism

To take advantage of parallelism during the restore, determine the number of CPUs on your server by executing the following:

```
[oracle@stbydb1]$ grep -c ^processor /proc/cpuinfo
20
```

Make the following RMAN configuration changes at the Standby.

The example uses 8 preconfigured channels for RMAN to use during the recovery process.

```
[oracle@stbydb1]$ rman target /
RMAN> CONFIGURE DEFAULT DEVICE TYPE TO DISK;
RMAN> CONFIGURE DEVICE TYPE DISK PARALLELISM 8;
```

16. Restore the Standby Database from the primary database service

To backup a single large file in parallel, RMAN's multi section backup/restore capability improves backup and recovery rates. Underneath the covers RMAN divides the work among multiple channels and each channel acts upon a file section in a file. If you specify a small section size that would produce more than 256 sections, then RMAN increases the section size to a value that results in exactly 256 sections.

Section size clause depends on various factor such as, network bandwidth, number of channels, sizes of data files and application datafile sizes.

```
[oracle@stbydb1]$ sqlplus system/welcome1@chicago
SQL> select TABLESPACE_NAME, bytes/1024/1024/1024 SIZE_IN_GB from dba_data_files;

TABLESPACE_NAME                SIZE_IN_GB
------------------------------ ----------
SYSTEM                          .68359375
SYSAUX                           .5859375
UNDOTBS1                       .297851563
UNDOTBS2                         .1953125
```

```
USERS                          .004882813
```

For example, the following command executed on the standby host specifies a backup section size of 1 GB.

```
[oracle@stbydb1]$ rman target /
RMAN> restore database from service chicago section size 1G;
RMAN> backup spfile;
```

17. Start managed recovery on the standby

```
[oracle@stbydb1]$ sqlplus / as sysdba
SQL> ALTER DATABASE RECOVER MANAGED STANDBY DATABASE DISCONNECT FROM SESSION;
```

18. Register the standby database with Clusterware

```
[oracle@stbydb1]$ export ORACLE_HOME=/u01/app/oracle/product/12.1.0.2/dbhome_1
[oracle@stbydb1]$ export ORACLE_SID=boston1
[oracle@stbydb1]$ export PATH=$ORACLE_HOME/bin:$PATH
```

Single instance example

```
[oracle@stbydb1]$ srvctl add database -db boston -oraclehome /u01/app/oracle/product/12.2.0.1/dbhome_1 -dbtype SINGLE -
instance boston1 -node stbydb1 -dbname chicago -acfspath
'/u01/app/oracle/oradata/datastore,/u02/app/oracle/oradata/datastore,/u01/app/oracle/fast_recovery_area/datastore' -role
physical_standby -spfile '/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston/spfileboston.ora' -pwfile
'/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston/orapwboston'
```

RAC example

```
[oracle@stbydb1]$ srvctl add database -db boston -oraclehome /u01/app/oracle/product/12.2.0.1/dbhome_1 -dbtype RAC -
dbname chicago -acfspath
'/u01/app/oracle/oradata/datastore,/u02/app/oracle/oradata/datastore,/u01/app/oracle/fast_recovery_area/datastore' -role
physical_standby -spfile '/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston/spfileboston.ora' -pwfile
'/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston/orapwboston'
[oracle@stbydb1]$ srvctl add instance -database boston -instance boston1 -node stbydb1
[oracle@stbydb1]$ srvctl add instance -database boston -instance boston2 -node stbydb2
[oracle@stbydb1]$ scp $ORACLE_HOME/dbs/initboston1.ora oracle@
stbydb2:/u01/app/oracle/product/12.2.0.1/dbhome_1/dbs/initboston2.ora
[oracle@stbydb1]$ srvctl add instance -database boston -instance boston1 -node stbydb1
[oracle@stbydb1]$ srvctl add instance -database boston -instance boston2 -node stbydb2
```

19. Set the parameters and create the Broker configuration.

Modify the script below to your environment and save as PostCR.sql

NOTE: Flashback database is required to re-instantiate a failed primary after a failover role transition. Optionally enable flashback on both the primary and standby. The standby database can begin using flashback on using the PostCR script as follows.

```
[oracle@stbydb1]$ cat PostCR.sql
connect / as sysdba
alter system set dg_broker_config_file1='/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston/dr1.dat' scope=both;
alter system set dg_broker_config_file2='/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston/dr2.dat' scope=both;
```

```
alter system set db_flashback_retention_target=120 scope=spfile;
alter database flashback on;
alter system set dg_broker_start=true scope=spfile;
shutdown immediate
startup mount
alter system register;
connect sys/welcome1@chicago as sysdba
alter system set
dg_broker_config_file1='/u02/app/oracle/oradata/datastore/.ACFS/snaps/chicago/chicago/dr1.dat'
scope=both;
alter system set
dg_broker_config_file2='/u02/app/oracle/oradata/datastore/.ACFS/snaps/chicago/chicago/dr2.dat' scope=both;
alter system set dg_broker_start=TRUE;
host sleep 30
host dgmgrl sys/welcome1@chicago "CREATE CONFIGURATION dgconfig AS PRIMARY
DATABASE IS CHICAGO CONNECT IDENTIFIER IS CHICAGO";
host sleep 30
host dgmgrl sys/welcome1@chicago "ADD DATABASE BOSTON AS CONNECT IDENTIFIER IS BOSTON" ;
host dgmgrl sys/welcome1@chicago "ENABLE CONFIGURATION"
exit
```

Execute the script PostCR.sql on the standby database. Set your environment to standby database

```
[oracle@stbydb1]$ export ORACLE_HOME=/u01/app/oracle/product/12.2.0.1/dbhome_1
[oracle@stbydb1]$ export ORACLE_SID=boston1
[oracle@stbydb1]$ export PATH=$ORACLE_HOME/bin:$PATH
[oracle@stbydb1]$ sqlplus / as sysdba
SQL> @PostCR.sql
```

20. Verification using sqlplus/srvctl

```
[oracle@stbydb1]$ srvctl config database -d chicago
[oracle@stbydb1]$ srvctl config database -d boston
[oracle@stbydb1]$ sqlplus / as sysdba
SQL> select FORCE_LOGGING, FLASHBACK_ON, OPEN_MODE, DATABASE_ROLE, SWITCHOVER_STATUS, DATAGUARD_BROKER,
PROTECTION_MODE from v$database;
SQL> select PROCESS,PID,DELAY_MINS from V$MANAGED_STANDBY;
```

21. Verification from dg broker (using dgmgrl)

```
$ dgmgrl
DGMGRL> connect sys/welcome1@boston
DGMGRL> show configuration verbose
DGMGRL> show database verbose chicago
DGMGRL> show database verbose boston
DGMGRL> validate database chicago
DGMGRL> validate database boston
```

22. Switchover tests

$ dgmgrl
DGMGRL> connect sys/welcome1@boston
DGMGRL> switchover to boston

DGMGRL> connect sys/welcome1@chicago
DGMGRL> switchover to chicago;

23. Failover tests

connect to standby before failover:

$ dgmgrl
DGMGRL> connect sys/welcome1@boston
DGMGRL> failover to boston
DGMGRL> reinstate database chicago

connect to former primary before failover:

DGMGRL> connect sys/welcome1@chicago
DGMGRL> failover to chicago;
DGMGRL> reinstate database boston

Appendix B: 11gR2 Example Setup on Oracle Database Appliance

Sample Environment

The following section describes the primary and standby database environment topologies used in the subsequent Data Guard setup example using Oracle Database Appliance.

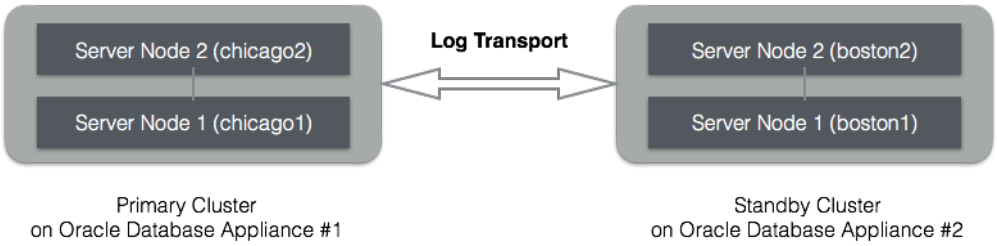

Figure 2 Configuration Topology of Oracle RAC on Oracle Database Appliance

	Primary Oracle Database Appliance		Standby Oracle Database Appliance	
Appliance Name	appliance#1		appliance#2	
Host Names	proddb1	proddb2	stbydb1	stbydb2
Cluster Name	PCLUSTER		SCLUSTER	
Database Name	chicago		chicago	
Database Unique Name	chicago		boston	
Instance Name	chicago1	chicago2	boston1	boston2
SCAN Name and IPs	proddb-scan (10.1.27.2, 10.1.27.3)		stbydb-scan (10.1.27.4, 10.1.27.5)	
Grid Infrastructure Software Installation	/u01/app/12.1.0.2/grid		/u01/app/12.1.0.2/grid	
Oracle Database Software Installation	/u01/app/oracle/product/11.2.0.4/db_home1		/u01/app/oracle/product/11.2.0.4/db_home1	
ARCHIVELOG mode	Yes		Yes	
FORCE LOGGING mode	Yes		Yes	

Table 1 - Example Oracle Database naming conventions

Primary Environment Configuration

1. Create Standby Redo Logs

Standby redo logs host redo data received from the primary database. In advance of the primary standby setup, Oracle recommends that standby redo logs be created on the primary database as well so that it is immediately ready to receive redo data following a switch-over to the standby role.

Create Standby Redo Logs (SRL) on the primary database. Each thread of the standby redo log must have at least one more redo log group than the corresponding thread of the online redo log. For example,

```
SQL> alter database add standby logfile thread 1 group 7 size 1G, group 8 size 1G, group 9 size 1G,
group 10 size 1G;
SQL> alter database add standby logfile thread 2 group 11 size 1G,group 12 size 1G, group 13 size 1G,
group 14 size 1G;
```

To check the number of online redo logs & their sizes, use the following query.

```
SQL> select thread#, group#, bytes/1024/1024/1024 SIZE_IN_GB, status from v$log;
```

Note that the size of the standby redo logs should match the size of the redo logs. The standby redo logs have to be created on the REDO disk group which resides on the solid state disks.

To validate the size of each log file and number of log groups in the standby redo log, use the following query.

```
SQL> select group#, thread#, bytes from v$standby_log;
```

2. Enable archivelog mode on primary database

Archiving is the process of saving and protecting REDO information in the form of archive files before the redo logs of an active database are overwritten in a circular manner. Database created on Oracle Database Appliance have archiving turned on by default. However, it is not mandatory to run your databases in archive log mode.

Verify that the primary database is running in ARCHIVELOG mode.

```
SQL> archive log list
```

If the primary database is not running in ARCHIVELOG mode, then enable ARCHIVELOG mode as follows.

Shutdown both instances on Oracle Database Appliance.

```
$ srvctl stop database –d chicago
```

Startup mount one instance in exclusive mode.

```
SQL> startup mount exclusive;
```

Turn on archiving.

```
SQL> alter database archivelog;
```

Shutdown the instance.

```
SQL> shutdown immediate;
```

Restart the database.

```
$ srvctl start database –d chicago
```

3. Enable FORCE LOGGING mode.

Force logging enables you to capture database operations performed with the NOLOGGING attribute. This ensures integrity of your standby database. Verify if FORCE LOGGING is already enabled on your primary database.

```
SQL> select force_logging from v$database;
```

If FORCE LOGGING is not enabled, then enable it using the following commands.

```
SQL> alter database force logging;
```

4. Configure Flashback Database feature

The Oracle Flashback Database feature provide a fast alternative to performing incomplete database recovery. Although using the Flashback Database feature is optional, it can be very useful for faster re-instatement of the old primary database after a failover. Thus, if you do a failover to the standby, and the old primary can be repaired, you do not have to rebuild the old primary database as a standby database but simply flashback and let Oracle Data Guard resynchronize from that point onwards.

Check if the primary database has Flashback Database enabled and if required enable it.

```
SQL> select flashback_on from v$database;
SQL> alter database flashback on;
```

Note that enabling Flashback Database will require additional space consumption in the Fast Recovery Area (RECO Disk Group). The space used by flashback logs can be controlled by setting the parameter DB_FLASHBACK_RETENTION_TARGET to a desired value. This value is specified in minutes. For example,

```
SQL> alter system set DB_FLASHBACK_RETENTION_TARGET=120 scope=both sid='*';
```

5. Enable Standby File Management

When the primary database adds or drops a datafile, the corresponding action should also be automatically taken on the standby database. This operation can be enabled using automated standby file management.

```
SQL> alter system set STANDBY_FILE_MANAGEMENT=AUTO scope=both sid='*';
```

6. Enable Remote Privileged Login

Ensure that each instance of the primary database is configured with remote login password file. Note that the Oracle Database Appliance deploys the databases with this setting. The initialization parameter REMOTE_LOGIN_PASSWORDFILE must be set to exclusive. If this parameter was reset in your environment and needs to be modified as below, it requires a database restart for it to take effect.

```
[oracle@proddb1]$ sqlplus / as sysdba
SQL> show parameter remote_login_passwordfile
SQL> alter system set remote_login_passwordfile='exclusive' scope=spfile sid='*';
```

7. Setup TNS Entries

Oracle Net Service Names must be configured to enable redo transportation across the databases. Update tnsnames.ora file to include the TNS alias for both primary and standby databases. Note that in the Oracle Database Appliance, the tnsnames.ora file is located in network/admin directory of the Oracle database home.

```
chicago =
 (DESCRIPTION =
  (ADDRESS = (PROTOCOL = TCP)(HOST = proddb-scan)(PORT = 1521))
  (CONNECT_DATA =
   (SERVER = DEDICATED)
   (SERVICE_NAME = chicago)
  )
 )
```

```
boston =
 (DESCRIPTION =
  (ADDRESS = (PROTOCOL = TCP)(HOST = stbydb-scan)(PORT = 1521))
  (CONNECT_DATA =
   (SERVER = DEDICATED)
   (SERVICE_NAME = boston)
  )
 )
```

8. Setup Redo Transport Service

The Oracle Data Guard redo transport mechanism uses Oracle Net connections to send the redo between the databases. Redo transport is enabled by setting the LOG_ARCHIVE_DEST_n parameter. For example, the following setup enables log shipping and uses LGWR based transmission in asynchronous mode.

```
SQL> alter system set log_archive_dest_2='SERVICE=boston LGWR ASYNC REGISTER VALID_FOR=(online_logfile,primary_role)
REOPEN=60 DB_UNIQUE_NAME=boston' scope=both sid='*';
```

More details about redo log transmission options can be found in Oracle Data Guard Concepts and Administration Guide.

9. Setup Fetch Archive Log Server

When the database is in standby role and the primary is unable to send any missing log files, then the standby database can use the FAL_SERVER setting to pull those missing log files. The FAL_SERVER parameter is uses the Oracle Net service name.

```
SQL> alter system set FAL_SERVER=boston scope=both sid='*';
```

Standby Environment Configuration

This section describes the steps that must be executed on the standby database. It is assumed that you have set up Oracle Database Appliance system in the standby environment. For setting up Oracle Database Appliance in a Bare Metal or Virtualized Platform configuration please refer to Oracle Database Appliance Setup Poster at http://docs.oracle.com/cd/E22693_01/doc.12/e55694.pdf

10. Setup TNS Entries

Oracle Net Service Names must be configured to enable redo transportation across the databases. Update tnsnames.ora file to include the TNS alias for both primary and standby databases. Note that on the Oracle Database Appliance, the tnsnames.ora file is located in network/admin directory of the Oracle database home.

```
chicago =
 (DESCRIPTION =
  (ADDRESS = (PROTOCOL = TCP)(HOST = proddb-scan)(PORT = 1521))
  (CONNECT_DATA =
   (SERVER = DEDICATED)
   (SERVICE_NAME = chicago)
  )
 )

boston =
 (DESCRIPTION =
  (ADDRESS = (PROTOCOL = TCP)(HOST = stbydb-scan)(PORT = 1521))
  (CONNECT_DATA =
   (SERVER = DEDICATED)
   (SERVICE_NAME = boston)
  )
 )
```

11. Create Static Listener Configuration

As the grid user, create a static listener service on the standby database for Recovery Manager (RMAN) connection during instantiation. Note that the listener home is in the Grid Infrastructure home (/u01/app/12.1.0.2/grid/network/admin)

```
SID_LIST_LISTENER =
    (SID_LIST = (SID_DESC = (GLOBAL_DBNAME = boston)
    (ORACLE_HOME = /u01/app/oracle/product/11.2.0.4/dbhome_1)(SID_NAME = boston)))
```

12. Restart Listener

After changes to the listener are made, it must be restarted.

```
[grid@stbydb1]$ lsnrctl reload listener
```

13. Create Initial Standby Parameter File

On the standby host in the ORACLE_HOME/dbs directory create a pfile (initboston1.ora) with the following parameters. It is recommended to set the sga_target same as that of the primary database. For example:

```
[oracle@stbydb1]$ vi /u01/app/oracle/product/11.2.0.4/dbhome_1/dbs/initboston.ora
db_name=chicago
db_unique_name=boston
sga_target=5G
```

14. Create Password File

During the RMAN duplication process, the auxiliary instance needs to be accessed with remote authentication that requires the creation of the password file.

```
[oracle@stbydb1]$ orapwd file=/u01/app/oracle/product/11.2.0.4/dbhome_1/dbs/orapwboston password=<primary sysdba passwd>
```

15. Create Audit Directory

Create audit file destination directory on the standby side on both nodes.

```
[oracle@stbydb1]$ mkdir -p /u01/app/oracle/admin/boston/adump
[oracle@stbydb2]$ mkdir -p /u01/app/oracle/admin/boston/adump
```

16. Create the needed ACFS storage directories on the standby Oracle Database Appliance system

Use the “oakcli create dbstorage” as ‘root’ user command to create ACFS database storage for the standby:

```
[root@stbydb1]# oakcli create dbstorage -db boston
```

17. Startup Standby Instance

Startup the standby database instance on first standby host in NOMOUNT state to prepare for instantiation.

```
[oracle@stbydb1]$ export ORACLE_SID=boston
[oracle@stbydb1]$ sqlplus / as sysdba
SQL> startup nomount
```

18. Validate Network Connectivity

At this stage, Oracle Net should be able to resolve the TNS aliases for both the primary and standby environments from the standby environment.

```
[oracle@stbydb1]$ tnsping chicago
[oracle@stbydb1]$ tnsping boston
[oracle@stbydb1]$ sqlplus sys/<password>@//proddb1:1521/chicago as sysdba
```

Instantiate Standby Database

This section outlines the instantiation of the standby database after the setup on the primary and standby environments is complete.

19. Duplicate Database

Using Oracle Recovery Manager (RMAN), the standby database can be created with DUPLICATE DATABASE command. As part of the duplication process, the parameter file, password file, controlfile, and database files are copied over from the primary environment to the standby environment.

The appropriate changes required to the parameter settings for standby operation also need to be specified in the RMAN DUPLICATE DATABASE command. Once RMAN copies over the primary parameter file, the parameters specified in the DUPLICATE DATABASE command are changed accordingly.

As the password file is also copied over, the standby database would have the same password as the primary database.

```
[oracle@stbydb1]$ mkdir -p /u01/app/oracle/oradata/datastore/boston/BOSTON/controlfile
[oracle@stbydb1]$ mkdir -p /u01/app/oracle/oradata/datastore/boston/BOSTON/onlinelog
[oracle@stbydb1]$ mkdir -p /u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/BOSTON/datafile
[oracle@stbydb1]$ mkdir -p /u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston
[oracle@stbydb1]$ mkdir -p /u01/app/oracle/oradata/datastore/boston/BOSTON
[oracle@stbydb1]$ mkdir -p /u01/app/oracle/fast_recovery_area/datastore/boston/BOSTON/archivelog
[oracle@stbydb1]$ mkdir -p /u01/app/oracle/fast_recovery_area/datastore/boston/BOSTON/flashback

[oracle@stbydb1]$ export NLS_DATE_FORMAT="HH24:MI:SS"
[oracle@stbydb1]$ rman
connect target sys/welcome1@//proddb1:1521/chicago
connect auxiliary sys/welcome1@//stbydb1:1521/boston
run {
allocate channel tgt1 device type disk;
allocate auxiliary channel aux1 device type disk;
allocate channel tgt2 device type disk;
allocate auxiliary channel aux2 device type disk;
allocate channel tgt3 device type disk;
allocate auxiliary channel aux3 device type disk;
DUPLICATE TARGET DATABASE FOR STANDBY FROM ACTIVE DATABASE
SPFILE
PARAMETER_VALUE_CONVERT='chicago','boston','CHICAGO','BOSTON'
SET DB_NAME="chicago"
SET DB_UNIQUE_NAME="boston"
SET CLUSTER_DATABASE='false'
SET REMOTE_LISTENER="stbydb-scan:1521"
SET LOCAL_LISTENER=""
SET STANDBY_FILE_MANAGEMENT='AUTO'
set audit_file_dest = '/u01/app/oracle/admin/boston/adump'
set db_create_file_dest = '/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston'
set diagnostic_dest="/u01/app/oracle"
set db_recovery_file_dest="/u01/app/oracle/fast_recovery_area/datastore"
set db_create_online_log_dest_1="/u01/app/oracle/oradata/datastore/boston"
set log_archive_dest_2 = 'service=chicago lgwr async register
valid_for=(online_logfiles, primary_role) db_unique_name=chicago'
set fal_server='chicago'
NOFILENAMECHECK;
}
```

Modify db_file_name_convert and log_file_name_convert parameters and restart the instance

```
[oracle@stbydb1]$ sqlplus / as sysdba
SQL> alter system set db_file_name_convert='chicago','boston','CHICAGO','BOSTON' scope=spfile;
SQL> alter system set log_file_name_convert='chicago','boston','CHICAGO','BOSTON' scope=spfile;
SQL> shutdown immediate
SQL> startup
```

Start Managed Recovery Mode

Start managed recovery on the standby database in real-time mode as follows:

```
SQL> alter database recover managed standby database disconnect;
```

With real-time apply, redo apply services can apply redo log to the standby database as soon as it is received without having to wait for the current standby redo log to be archived. This results in faster failover and switchover times because the standby redo log files have already been applied by the time a failover or switchover occurs.

20. Enable Flashback Database

Enable Flashback Database on the standby database and adjust retention as required. For example,

```
SQL> alter database recover managed standby database cancel;
SQL> alter database flashback on;
SQL> alter system set DB_FLASHBACK_RETENTION_TARGET=120;
SQL> alter database recover managed standby database disconnect;
```

Post Instantiation Steps

The following steps are performed after the standby instantiation has been completed.

21. Register standby database with Oracle Clusterware.

Make sure that the ORACLE_HOME environment variable is set correctly. Register the standby database with Oracle Clusterware as single instance to run from one node of the cluster.

```
[oracle@stbydb1]$ mv /u01/app/oracle/product/11.2.0.4/dbhome_1/dbs/spfileboston.ora
/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston/spfileboston.ora
[oracle@stbydb1]$ echo "spfile='/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston/spfileboston.ora'" >
/u01/app/oracle/product/11.2.0.4/dbhome_1/dbs/initboston1.ora
```

For a non-Active Data Guard configuration,

```
[oracle@stbydb1]$ srvctl add database –d boston –o /u01/app/oracle/product/11.2.0.4/dbhome_1 -p
/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston/spfileboston.ora –r physical_standby –s mount -c SINGLE -x
stbydb1 –j
'/u01/app/oracle/oradata/datastore,/u02/app/oracle/oradata/datastore,/u01/app/oracle/fast_recovery_area/datastore'
```

For a Active Data Guard configuration,

```
[oracle@stbydb1]$ srvctl add database –d boston –o /u01/app/oracle/product/11.2.0.4/dbhome_1 –p
/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston/spfileboston.ora –r physical_standby –s 'read only' -c
SINGLE -x stbydb1 –j
'/u01/app/oracle/oradata/datastore,/u02/app/oracle/oradata/datastore,/u01/app/oracle/fast_recovery_area/datastore'
```

Start up the database resource

```
[oracle@stbydb1]$ sqlplus "/ as sysdba"
SQL> shutdown immediate
```

22. Convert the standby database to Oracle RAC

This step is optional. At this stage the standby database is configured as a single instance database. If the primary database was RAC database, the standby database can also be converted into RAC standby. Appendix C provides information on using the *rconfig* tool to convert the single instance database to RAC standby database.

23. Setup Dedicated DR Network

This step is optional. The Redo Transport Services can be configured to use a dedicated network. A dedicated network channel can help in improving the performance of redo transmission especially when the application network traffic consumes most of available bandwidth on the public network. Please refer to MOS note 1451810.1, Configuring Dedicated Disaster Recovery Network on Oracle Database Appliance, for more information on setting up a dedicated network channel for Data Guard Redo Transport.

24. Verify Configuration and Setup

On the standby database internal data dictionary views can be used to verify standby database operations.

```
[oracle@stbydb1]$ srvctl config database -d boston
SQL> select database_role, switchover_status from v$database;
SQL> select thread#, sequence#, applied from v$archived_log order by sequence#;
```

Creating Data Guard Broker Configuration

This step is optional. Creating a Data Guard Broker configuration enables easier management of the entire Data Guard environment as a single entity. It provides management, maintenance and monitoring capabilities that can be used both locally and remotely.

25. Configure listeners for static registration

Configure listeners for static registration of all the instances of primary & standby databases. In the Oracle Database Appliance, the listeners are running from the Grid Infrastructure home. An example of static registration for a RAC primary & standby configuration:

On node proddb1:

```
SID_LIST_LISTENER =
    (SID_LIST =
    (SID_DESC = (GLOBAL_DBNAME = chicago_DGMGRL)
    (ORACLE_HOME = /u01/app/oracle/product/11.2.0.4/dbhome_1)
    (SID_NAME = chicago1)))
```

On node proddb2:

```
SID_LIST_LISTENER =
    (SID_LIST =
    (SID_DESC = (GLOBAL_DBNAME = chicago_DGMGRL)
    (ORACLE_HOME = /u01/app/oracle/product/11.2.0.4/dbhome_1)
    (SID_NAME = chicago2)))
```

On node stbydb1:

```
SID_LIST_LISTENER =
    (SID_LIST =
    (SID_DESC =
    (GLOBAL_DBNAME = boston_DGMGRL)
    (ORACLE_HOME = /u01/app/oracle/product/11.2.0.4/dbhome_1)
    (SID_NAME = boston1)))
```

On node stbydb2:

```
SID_LIST_LISTENER =
    (SID_LIST =
    (SID_DESC = (GLOBAL_DBNAME = boston_DGMGRL)
    (ORACLE_HOME = /u01/app/oracle/product/11.2.0.4/dbhome_1)
    (SID_NAME = boston2)))
```

26. Configure Broker Configuration Files

Configure location of broker configuration files at both primary and standby databases.

```
[oracle@proddb1]$ mkdir –p /u02/app/oracle/oradata/datastore/chicago/broker
SQL> ALTER SYSTEM SET DG_BROKER_CONFIG_FILE1='/u02/app/oracle/oradata/datastore/chicago/broker/dr1.dat'
SCOPE=BOTH SID='*';
SQL> ALTER SYSTEM SET DG_BROKER_CONFIG_FILE2='/u02/app/oracle/oradata/datastore/chicago/broker/dr2.dat'
SCOPE=BOTH SID='*';

[oracle@stbydb1]$ mkdir -p /u02/app/oracle/oradata/datastore/boston/broker
SQL> ALTER SYSTEM SET DG_BROKER_CONFIG_FILE1='/u02/app/oracle/oradata/datastore/boston/broker/dr1.dat'
SCOPE=BOTH SID='*';
SQL> ALTER SYSTEM SET DG_BROKER_CONFIG_FILE2='/u02/app/oracle/oradata/datastore/boston/broker/dr2.dat'
SCOPE=BOTH SID='*';
```

27. Enable Data Guard Broker

Enable Data Guard Broker on both primary and standby databases.

On node proddb1:

```
SQL> ALTER SYSTEM SET DG_BROKER_START=TRUE SCOPE=BOTH SID='*';
```

On node stbydb1:

```
SQL> ALTER SYSTEM SET DG_BROKER_START=TRUE SCOPE=BOTH SID='*';
```

28. Create Broker Configuration

Create the broker configuration on the primary using the DB_UNIQUE_NAME of the primary database and its corresponding TNS alias.

```
[oracle@proddb1]$ dgmgrl
DGMGRL> connect sys/welcome1;
DGMGRL> CREATE CONFIGURATION 'ODADGConfig' AS PRIMARY DATABASE IS CHICAGO CONNECT IDENTIFIER is CHICAGO;
```

29. Add Standby Database to Data Guard Broker Configuration

Add standby database to the configuration using the DB_UNIQUE_NAME of the standby database.

```
DGMGRL> ADD DATABASE BOSTON AS CONNECT IDENTIFIER IS BOSTON;
```

30. Enable Configuration

Enable Data Guard Broker configuration as follows.

```
DGMGRL> ENABLE CONFIGURATION;
```

31. Check configuration

Run the following command to verify the established configuration.

```
DGMGRL> show configuration;
DGMGRL> show database verbose chicago;
DGMGRL> show instance verbose chicago1;
DGMGRL> show instance verbose chicago2;
DGMGRL> show database verbose boston;
DGMGRL> show instance verbose boston1;
DGMGRL> show instance verbose boston2;
```

32. Switchover tests

```
$ dgmgrl
DGMGRL> connect sys/welcome1@boston
DGMGRL> switchover to boston
```

DGMGRL> connect sys/welcome1@chicago
Connected as SYSDBA.
DGMGRL> switchover to chicago;

33. Failover tests

connect to standby before failover:

$ dgmgrl
DGMGRL> connect sys/welcome1@boston
DGMGRL> failover to boston
DGMGRL> reinstate database chicago

connect to former primary before failover:

DGMGRL> connect sys/welcome1@chicago
DGMGRL> failover to chicago;
DGMGRL> reinstate database boston

Appendix C: Converting Single Instance Databases to Oracle RAC

You can use the *rconfig* command line utility to convert a single-instance database to an Oracle RAC database, or to convert it to an Oracle RAC One Node.

To use this feature, complete the following steps:

1. Create Configuration XML File

A sample of the configuration XML file to be saved as convert.xml is shown below. You may modify this file as required for your system.

The sample XML files are in $ORACLE_HOME/assistants/rconfig/sampleXML directory.

Note: Set the convert option Convert verify="ONLY" initially to perform a test conversion to ensure that a conversion can be completed successfully.

```
<?xml version="1.0" encoding="UTF-8"?>
<n:RConfig xmlns:n="http://www.oracle.com/rconfig"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.oracle.com/rconfig rconfig.xsd">
  <n:ConvertToRAC>
          <n:Convert verify="YES">
             <n:SourceDBHome>/u01/app/oracle/product/11.2.0.4/dbhome_1</n:SourceDBHome>
             <n:TargetDBHome>/u01/app/oracle/product/11.2.0.4/dbhome_1</n:TargetDBHome>
       <n:SourceDBInfo SID="boston">
              <n:Credentials>
               <n:User>sys</n:User>
               <n:Password>welcome1</n:Password>
               <n:Role>sysdba</n:Role>
              </n:Credentials>
             </n:SourceDBInfo>
             <n:NodeList>
              <n:Node name="stbydb1"/>
              <n:Node name="stbydb2"/>
             </n:NodeList>
             <n:InstancePrefix>stbydb</n:InstancePrefix>
       <n:SharedStorage type="CFS">
         <n:TargetDatabaseArea></n:TargetDatabaseArea>
         <n:TargetFlashRecoveryArea></n:TargetFlashRecoveryArea>
       </n:SharedStorage>
    </n:Convert>
  </n:ConvertToRAC>
</n:RConfig>
```

2. Run rconfig Tool

When you have completed making changes, save the file. Run the following command on the standby database. The *convert.xml* is the name of the XML input file you configured above.

```
[oracle@stbydb1]$ rconfig convert.xml
```

3. Update Cluster Ready Services Resource

The Cluster Ready Services (CRS) resource must be updated for the converted database.

```
[oracle@stbydb1]$ srvctl modify database -d boston -r physical_standby -s mount
```

Restart the standby database

```
[oracle@stbydb1]$ srvctl stop database –d boston
[oracle@stbydb1]$ srvctl start database –d boston
```

4. Validate Configuration

Validate the configuration of standby database.

```
[oracle@stbydb1]$ srvctl config database -d boston
```

Appendix D: Upgrading Database with Oracle Data Guard

Upgrading SERVER, STORAGE, and DATABASE components

Upgrading an ODA environment consists of upgrading the SERVER, STORAGE, and DATABASE components. When upgrading an ODA environment where a standby system is already implemented, you can leverage the standby system to reduce downtime required for completing the upgrade activities. The purpose of this section is to provide a high-level overview of the upgrade process in a primary-standby setup.

1. Verify system is operating normally (run pre-checks, validate hardware and system processes (oakd, etc.), verify system configuration using orachk, etc.)
2. Take a backup of the OS, GI, and Oracle homes, and databases (in the primary environment)
3. Upgrade SERVER and GI components on the standby ODA system
4. Switchover primary database role and application connections to the standby ODA system
5. Upgrade SERVER and GI components on the original primary ODA system
6. Create new DB home on the current standby with the desired version using "oakcli create dbhome" command, if one does not exist or you do not want to use an existing home
7. Create new DB home with the desired version on the current primary, if one does not exist or you do not want to use an existing home
8. Defer redo transfer from primary to standby
9. Run "oakcli upgrade database" command on the current standby database (only binaries would be upgraded/switched and catalog scripts will not be run)
10. Stop the application traffic
11. Upgrade current primary database using "oakcli upgrade database" command
12. Start the application traffic
13. Enable redo transfer from primary to the standby database
14. Verify DB operations on primary and standby side
15. Optionally switchback roles between primary and standby environments

Using the above process, the downtime during the upgrade is minimized and system availability is affected for only the duration of the upgrade of the database component.

Upgrading only the DATABASE component

If upgrading SERVER (OS, GI, general firmware) and STORAGE (firmware on shared disks) components, leveraging switchover/switchback can help reduce downtime. However, if you are only upgrading the DATABASE component, then unless you are using a zero downtime solution (such as active-active GoldenGate solution), some downtime is expected for the application. The general process to execute the database upgrade when a standby configuration exists may be outlined something like as follows.

1. Verify system is operating normally (run pre-checks, validate hardware and system processes (oakd, etc.), verify system configuration using orachk, etc.)
2. Take a backup of the database and Oracle homes
3. Create new DB home on the standby with the desired version using "oakcli create dbhome" command, if one does not exist or you do not want to use an existing home
4. Create new DB home with the desired version on the primary, if one does not exist or you do not want to use an existing home
5. Stop the application
6. Let standby DB sync up with primary and verify last SCN generated on primary is applied on the standby
7. Defer redo transfer from primary to standby database (optional; stop redo flow)
8. Run "oakcli upgrade database" command on the standby database (only binaries would be upgraded/switched and catalog scripts will not be run)
9. Upgrade primary database using "oakcli upgrade database" command
10. Start the application
11. Enable redo transfer from primary to the standby database
12. Verify DB operations on primary and standby side

The total downtime requirement is for the duration of the DB upgrade. A switchover and switchback is not required during database upgrade.

References

Oracle Database Appliance Website on OTN

http://www.oracle.com/technetwork/server-storage/engineered-systems/database-appliance/index.html

Oracle Maximum Availability Architecture Best Practices

http://www.oracle.com/technetwork/database/features/availability/oracle-database-maa-best-practices-155386.html

Oracle Real Application Clusters Website on OTN

http://www.oracle.com/technetwork/database/clustering/overview/index.html

Oracle Clusterware Website on OTN

http://www.oracle.com/technetwork/database/clusterware/overview/index.html

Oracle Data Guard Website on OTN

http://www.oracle.com/technetwork/database/features/availability/dataguardoverview-083155.html

Oracle Data Guard Concepts and Administration 12c Release 1

https://docs.oracle.com/database/122/SBYDB/title.htm

http://docs.oracle.com/database/121/SBYDB/toc.htm

My Oracle Support (MOS) Knowledge Content Notes

https://support.oracle.com/CSP/main/article?cmd=show&type=NOT&id=1075908.1

https://support.oracle.com/CSP/main/article?cmd=show&type=NOT&id=2275154.1

Oracle Database High Availability Website on OTN

http://www.oracle.com/technetwork/database/features/availability/index.html

Oracle Corporation, World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065, USA

Worldwide Inquiries
Phone: +1.650.506.7000
Fax: +1.650.506.7200

ORACLE

CONNECT WITH US

blogs.oracle.com/oracle
facebook.com/oracle
twitter.com/oracle
oracle.com

Integrated Cloud Applications & Platform Services

Oracle Database Appliance: Implementing Disaster Recovery Solutions Using Oracle Data Guard
Date: August 2017
Author: Ramasubramanian Athmanathan, Ravi Sharma, Krisztian Fekete, Sanjay Singh
Contributing Authors: Oracle RAC Pack and MAA Team