Data Guard Configuration And Operation

Author: **G S Chapman**

Date: **18th December 2007**

Version: **1.3**

Location of Document:

DOCUMENT HISTORY

Version	Date	Changed By:	Remarks
1.0	14/02/06	G S Chapman	Original Version
1.1	18/03/06	G S Chapman	Minor revisions.
1.2	29/11/07	G S Chapman	Minor revisions, some added information from Metalink used in latest recovery situation.
1.3	18/12/07	G S Chapman	Read Only and UAT information.

DOCUMENT DISTRIBUTION

Copy No	Name	Role	Organisation

DOCUMENT REFERENCES

Document Name	Originator	Part Number	Version	Date
Data Guard Concepts and Administration Guide.	Oracle Corporation			

TABLE OF CONTENTS

1 Data Guard Architecture and Concepts 1
1.1 Glossary 1
1.2 Architecture 1
1.2.1 Log Transport Services 1
1.2.2 Log Apply Services 1
1.2.3 Data Guard Broker 2
1.3 How it Works 2
1.4 Archiving Options 2
1.4.1 Archiving Process 2
1.4.2 Synchronous verses Asynchronous 2
2 Design 3
2.1 Data Guard Mode 3
2.2 Data Loss 3
2.2.1 Enforced Log Switching 3
2.3 Components 3
3 Pre Requisites 4
3.1 Hardware 4
3.2 Network 4
3.3 Software 4
4 Creating a Standby Database Environment 5
4.1 Assumptions 5
4.2 Procedure 5
4.2.1 Configure Primary and Standby Sites 5
4.2.2 Install Oracle Software on Each site. 6
4.2.3 Configure Listener on Each Site 6
4.2.4 Configure TNS entries on each site. 6
4.2.5 Create Initialization Files (Primary/Standby) 7
4.2.6 Backup the Primary database. 9
4.2.7 Create a Standby Control File 10
4.2.8 Create the Standby Site from the Backup Database 11
4.2.9 Create a password file for the Standby Site 11
4.2.10 Start the standby database 11
4.2.11 Recovery of Standby database 11
4.2.12 Enable Log Transport/Apply Services 12
4.2.13 Disable Log Transport/Apply Services 12
4.3 Temporary Files 13
4.4 Forced Logging 13
5 Data Guard Options 14
5.1 Standby Redo Logs 14
5.1.1 How to create Standby Redo Logs 14
5.1.2 Limitations to Standby Redo Logs 15
5.1.3 Differences in the Log Apply Services when using Standby Redo Logs 15
5.2 File Name conversion 15
5.2.1 `DB_FILE_NAME_CONVERT parameter` 15
5.2.2 Manual Renaming Files 16
5.2.3 LOG_FILE_NAME_CONVERT parameter 17
6 Enterprise Manager/Grid Control 19
6.1 Accessing Data Guard Manager 19
6.2 Adding an existing standby database 20

6.2.1 Select an existing standby database. 21
6.2.2 Set the standby archive location setting. 22
6.2.3 Step 3 Review the configuration settings. 22
7 Failover 23
7.1 Forced Failover – No Redo Logs 23
7.2 Forced Failover – Redo logs 23
7.3 Graceful failover 24
8 Using RMAN to Create the Standby Database 25
8.1 Assumptions: 25
8.2 RMAN Backup 25
8.2.1 New Backup 25
8.2.2 Existing Backup 25
8.3 Creating the Standby Database 26
8.3.1 Pre-requisites 26
8.3.2 Procedure 26
9 Changing Standby to Read ONLY mode. 27
9.1 Archivelogs from Primary in READ ONLY mode 27
9.2 Opening the database in READ ONLY mode 27
9.2.1 Error in alert log. 27
9.3 From Read Only to Managed Recovery Mode 28
10 Usage for UAT environments 29
10.1 Preparation 29
10.2 Backup standby database 29
10.3 Fully open standby database 30
10.4 User system testing 31
10.5 Post backups 31
10.6 Restore pre-backups 31
10.7 Restoring archivelogs using RMAN 32
10.8 Restart replication from primary db 32
11 Database patching/upgrading 34
11.1 Patch Set Apply Procedure 34
12 Monitoring Standby Database 36
12.1 Network Failure Preventing the Archiving of Logs to the Standby Site 36
12.2 Archived logs not applied on a physical standby database 36
12.2.1 To identify the logs in the archive gap 36
12.3 Manually Transmitting the Logs in the Archive Gap to the Standby Site 37
12.3.1 To copy logs in an archive gap to the standby site 37
12.4 Potential data loss window for a physical standby database 38
12.5 Determining the amount of redo sent to the remote destination 39
13 Considerations for changing primary 40
13.1 Temporary Tablespace 40
13.2 Redo Logs definitions and layout 40
13.3 Directory definitions 40
13.4 System file directory mappings. 40
13.5 Database Links 40
14 Problems encountered. 41
14.1 ORA-16009 Error and RFS Warnings 41

14.2 Synchronisation due to log transfer gap 41
14.3 ORA-16166 LGWR timed out on Network Server 0 45
A. Appendices 47
A.1 Sample Scripts 47
A.2 Large databases 47
A.3 Script to remove applied logs on standby system. 48
A.4 Monitoring scripts 49
A.4.1 Data Guard Physical Standby Diagnostic Information 49
A.4.2 Data Guard Primary Site Diagnostic Information 51
A.4.3 Alternative Standby Monitoring script 53

TABLE OF FIGURES

Figure 1 - Architectural Components 1
Figure 2 - 10G EM Database display 20
Figure 3 - 10g EM Standby Addition 21
Figure 4 - 10G EM adding existing Standby DB 21
Figure 5 - 10G EM Setting archive location 22
Figure 6 - 10G EM Configuration review 22

TABLES

Table 1 - Glossary 1
Table 2 - Site specific settings 3
Table 3 - Primary site intialization parameters 8
Table 4 - Standby site initialization parameters 9

PURPOSE OF DOCUMENT

This document is designed to provide a basic knowledge of the understanding of this system works. The document tries to go through Data Guard in detail in an effort to provide a better understanding of the product.

This document covers the processes, which are needed to facilitate the use of a standby database from both the perspective of the primary and contingency servers.

Currently this document will cover the following configuration:

Data Guard will be configured using the Maximum Availability model.

1 Data Guard Architecture and Concepts

1.1 Glossary

Primary Site	The Master database site. This is where the users connect to access the database.
Standby Site	The standby site is the disaster recovery site. Users will only connect to this database in the event of a failover.
Disaster	Non-availability of the primary site.

Table 1 - Glossary

1.2 Architecture

The Diagram below shows the Data Guard architectural components. These are explained in more detail below:

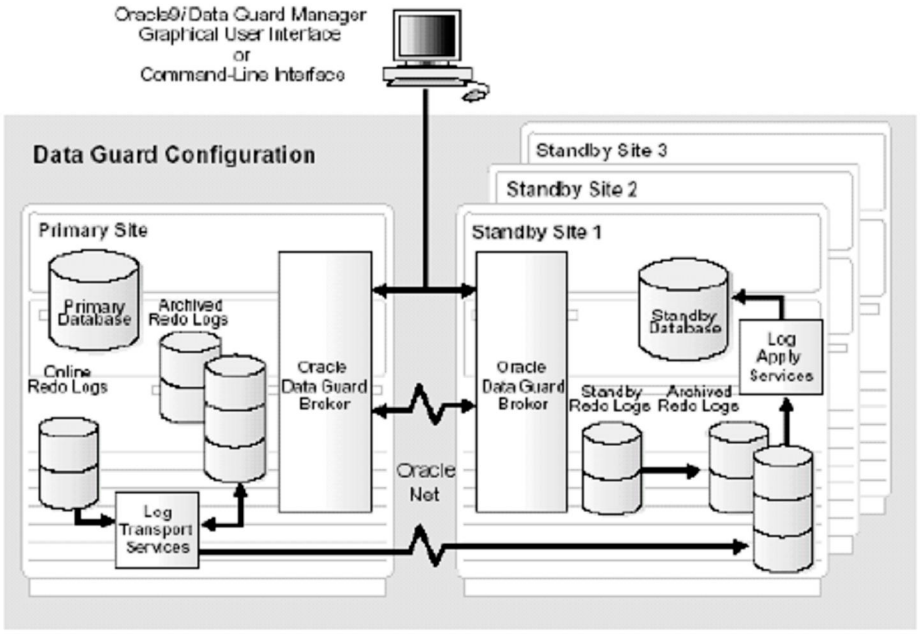

Figure 1 - Architectural Components

1.2.1 Log Transport Services

The Log transport services are designed to propagate changes from the primary database to the standby.

Log transport services provide control of different log archiving mechanisms; log archiving, error handling, reporting, and re-archiving logs after a system failure.

1.2.2 Log Apply Services

Logs apply services take the logs supplied by the Log transport services and apply them to the standby site.

1.2.3 Data Guard Broker

Data Guard broker is the management and monitoring component that helps create, control, and monitor a primary database protected by one or more physical standby databases. The broker automates the previously manual process of configuration.

Once it has created the Data Guard configuration, the broker monitors the activity, health, and availability for all systems in the Data Guard configuration.

Note: The easiest way to configure the Data Guard broker is to use the Data Guard Manager to discover the existing configuration.

1.3 How it Works

As information is written to the primary database this information is archived to the archive redo logs. When an online redo log is archived a copy of the archived redo log is sent to the standby site and applied to a database, which is acting in a standby capacity.

1.4 Archiving Options

1.4.1 Archiving Process

Traditionally all archiving activity has been conducted using the database archiver (ARCH) process. This is still the option most likely to be used.

In an environment where No data loss is a requirement then LGWR can be used to simultaneously write only redo log information to a remote destination.

Standard disaster recovery scenarios will continue to use the ARCH process, which is the default.

1.4.2 Synchronous verses Asynchronous

It is possible to specify whether archiving to the standby site is synchronous or asynchronous. Generally Asynchronous communication will only be used where network latency is slow.

When running in synchronous mode control will not be returned to the user until all synchronous destinations have been written too. i.e. If the ARCH process is archiving a redo log, the redo log will not be released back into the pool until all synchronous standby destinations have been written too. This is the default.

If using asynchronous communication, then archiving is undertaken in the background with control being returned to the user immediately.

Synchronous communication has the highest amount of data resilience on the standby site, but also has the greatest impact on performance.

2 Design

2.1 Data Guard Mode

Archive logs once generated on the primary database are transferred to the contingency machine and replayed against the standby database until the standby database is up to date. This method of update is known as the maximum performance model.

Other models include Maximum protection and Maximum Availability. These modes rely upon the simultaneous generation of online redo on both the primary and contingency servers.

One common decision is to use the maximum availability model. This model is a good compromise between network overhead and potential data loss.

2.2 Data Loss

Using the maximum performance model there is the potential for some data loss. This data loss can result when a primary database becomes unavailable and all of the information in the online redo logs is not flushed to an archive log.

A process needs to be put in place therefore which ensures that archive logs are forced at given intervals if they have not occurred automatically. A time period of 15 minutes is often chosen as the maximum log switch interval. This is based on the need to keep data loss to a maximum of 15 minutes, and the ability to maintain performance. (Redo logs should switch every 15 – 30 minutes to maintain optimal performance).

The method of protection chosen to be demonstrated is maximum availability. Whilst this reduces significantly it is possible under certain circumstances for some data to be lost. This generally occurs if the network link becomes unavailable.

2.2.1 Enforced Log Switching

2.2.1.1 Purpose

Enforced log switching is the process of forcing the database to switch logs if the last log switch was greater than 15 minutes ago.

2.2.1.2 How

This mechanism involves the setting of the database initialization parameter -archive_lag_target.

2.3 Components

Component	Value
Primary Host	macmain
Contingency Host	macback
Oracle SID	DGTEST
Oracle_Domain	macrotone.co.uk

Table 2 - Site specific settings

3 Pre Requisites

3.1 Hardware

The standby site will ideally be hosted by similar specification hardware to the primary site. Whilst this is not mandatory it ensures that if a failover to the standby site is required that similar levels of performance will be attained.

If lower specification hardware is being used then this will produce less performance/resilience than the main site. However it is envisaged that in this situation that the site would only be used in the case of disaster.

3.2 Network

In order to facilitate standby site, the primary database will transport archive redo logs to the standby site. It is therefore essential that the network link between the two sites:

- Is reliable (No single point of failure).
- Has suitable bandwidth (Depends on the amount of expected redo).
- Ideally has very low latency.

3.3 Software

Both the primary and standby sites are required to run the same version of the operating system and database software.

4 Creating a Standby Database Environment

4.1 Assumptions

For the purpose of the instructions below the following have been assumed:

- Primary Host Name is macmain
- Standby Host Name is macback
- Database Name is DGTEST.

Where these names are used for examples below they will be highlighted as above.

4.2 Procedure

The procedure to create a standby database environment is summarised below. Further sections go into detail about how to perform each of the following tasks:

1. Configure Primary and Standby Sites
2. Install Oracle Software on Each site.
3. Configure Listener on Each Site
4. Configure TNS on Each site.
5. Create Initialization Files (Primary/Standby).
6. Backup the Primary database.
7. Create a Standby Control File
8. Create the Standby Site from the Backup Database.
9. Create a password file for the standby database.
10. Start the standby database.
11. Enable Log Transport/Apply Services
12. Monitor.

4.2.1 Configure Primary and Standby Sites

To make management of the environment simpler (and the configuration of data guard), it is assumed that the Primary and Standby sites have exactly the same structure i.e.

- ORACLE_HOME points to the same mount point on both machines.
- ORACLE_BASE/admin points to the same mount point on both machines.
- The database name is the same on both machines.
- The location of the data files is the same on both machines.

Whilst this is not a mandatory requirement it is strongly advised because:

- Data Guard does not have to be configured to modify file locations.
- The initialization files (init.ora/spfile) can be the same in both environments.
- Any user-defined scripts can be the same in both environments.

4.2.2 Install Oracle Software on Each site.

The Oracle software will be installed from the Oracle Media on both sites.

4.2.3 Configure Listener on Each Site

Each of the sites will have an entry for the database statically included in the listener.ora file.

4.2.3.1 Primary Site Listener

```
LISTENER =
        (DESCRIPTION_LIST =
                (DESCRIPTION =
                        (ADDRESS_LIST =
                                (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC))
                        )
                        (ADDRESS_LIST =
                                (ADDRESS = (PROTOCOL = TCP)(HOST = macmain)(PORT = 1521))
                        )
                )
        )


SID_LIST_LISTENER =
        (SID_LIST =
                (SID_DESC =
                        (ORACLE_HOME =/server1/app/oracle/product/9.2.0)
                        (SID_NAME = DGTEST)
                )
    )
```

4.2.3.2 Standby Site Listener

```
LISTENER =
        (DESCRIPTION_LIST =
                (DESCRIPTION =
                        (ADDRESS_LIST =
                                (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC))
                        )
                        (ADDRESS_LIST =
                                (ADDRESS = (PROTOCOL = TCP)(HOST = macback)(PORT = 1521) )
                        )
                )
        )

SID_LIST_LISTENER =
        (SID_LIST =
                (SID_DESC =
                        (ORACLE_HOME = /server2/app/oracle/product/9.2.0)
                        (SID_NAME = DGTEST)
                )
        )
```

4.2.4 Configure TNS entries on each site.

In order to make things simpler the same Network service names will be generated on each site. These service names will be called:

Sample entries are:

4.2.4.1 Primary Site

```
standby_DGTEST.macrotone.co.uk =
        (DESCRIPTION =
                (ADDRESS_LIST =
                        (ADDRESS = (PROTOCOL = TCP)(HOST = macback)(PORT = 1521))
                )
                (CONNECT_DATA =
                        (ORACLE_SID = DGTEST)
                )
        )
```

```
        local_DGTEST.macrotone.co.uk =
                (DESCRIPTION =
                        (ADDRESS_LIST =
                                (ADDRESS = (PROTOCOL = TCP)(HOST =macmain)(PORT = 1521))
                        )
                        (CONNECT_DATA =
                                (SERVICE_NAME= DGTEST.macrotone.co.uk)
                        )
                )

        primary_DGTEST.macrotone.co.uk =
                (DESCRIPTION =
                        (ADDRESS_LIST =
                                (ADDRESS = (PROTOCOL = TCP)(HOST = macback)(PORT = 1521))
                        )
                        (CONNECT_DATA =
                                (SERVICE_NAME = DGTEST.macrotone.co.uk)
                        )
                )
```

4.2.4.2 Standby Site

```
standby_DGTEST.macrotone.co.uk =
        (DESCRIPTION =
                (ADDRESS_LIST =
                        (ADDRESS = (PROTOCOL = TCP)(HOST = macmain)(PORT = 1521))
                )
                (CONNECT_DATA =
                        (ORACLE_SID = DGTEST)
                )
        )
local_DGTEST.macrotone.co.uk =
        (DESCRIPTION =
                (ADDRESS_LIST =
                        (ADDRESS = (PROTOCOL = TCP)(HOST =macback)(PORT = 1521))
                )
                (CONNECT_DATA =
                        (SERVICE_NAME= DGTEST.macrotone.co.uk)
                )
        )
primary_DGTEST.macrotone.co.uk =
        (DESCRIPTION =
                (ADDRESS_LIST =
                        (ADDRESS = (PROTOCOL = TCP)(HOST = macmain)(PORT = 1521))
                )
                (CONNECT_DATA =
                        (SERVICE_NAME = DGTEST.macrotone.co.uk)
                )
        )
```

4.2.5 Create Initialization Files (Primary/Standby).

Certain initialization parameters are only applicable to one of the sites. Defining ALL of the parameters on both sites will ensure that if the roles are switched (Primary becomes Standby and Standby becomes the Primary) then no further configuration will be necessary.

4.2.5.1 Primary Site Initialization Parameters

The following Initialisation parameters need to be set on the primary site:

Parameter	Description
db_block_checksum	To enable database integrity checking (Advisable)
db_block_checking	To enable database integrity checking (Advisable)
log_archive_start	Start database creating archive logs. Note: This parameter is obsolete in Oracle 10g.
Remote_archive_enable	Enable remote archive log propagation.

Parameter	Description
log_archive_dest_1	This is the local destination for archive redo logs. This location must be mandatory.
Log_archive_dest_state_1	Set this parameter to enable to activate archive destination 1.
Log_archive_dest_2	This is the parameter, which determines how the standby database will function. This parameter will include the following values: SERVICE=standby_DGTEST.macrotone.co.uk(Net Service for Standby DB). In addition the following will be used: ARCH – Archive process with transmit the archive logs (Used in Max performance Mode). LGWR – LGWR process with transmit the archive logs (Used in Max Availability Mode). SYNC/ASYNC – Whether or not the standby database, will be written to synchronously or asynchronously. (Used in Max Availability Mode). AFFIRM – Ensures that Redo has been sent to the standby site before continuing with the next operation (Used in Max Availability Mode). MANDATORY/OPTIONAL – Whether or not it is mandatory for the standby site to be archived to at the same time as the primary. If mandatory and the standby is unavailable then the primary will stall until the solution is remedied. If optional and the standby is unavailable the primary site will continue. The FAL server will pick up any missing logs when communication with the standby is re-established. DELAY=60 When the standby database is in managed recovery mode, redo logs are automatically applied when they arrive from the primary database. However, to protect against the transfer of corrupted or erroneous data from the primary site to the standby site, you may want to create a time lag between archiving a redo log at the primary site and applying that archived redo log at the standby site. Use the DELAY attribute of the LOG_ARCHIVE_DEST_n initialization parameters to specify a time lag for the application of redo logs at the standby site. The DELAY attribute does not affect the transmittal of redo logs to the standby site. The DELAY attribute indicates that the archived redo logs at the standby site are not available for recovery until the specified time interval has expired. The time interval is expressed in minutes, and it starts when the redo log is successfully transmitted and archived at the standby site. If you specify the DELAY attribute without a time interval, the default time interval is 30 minutes. Typical Values Maximum Performance SERVICE=standby_DGTEST ARCH OPTIONAL Maximum Availability SERVICE=standby_DGTEST LGWR ASYNC AFFIRM OPTIONAL
Log_archive_dest_state_2	Set this parameter to enable to activate archive destination 2.
Fast_start_mttr_target	Set to 3600 or appropriate value.
Archive_lag_target	Maximum amount of time between log switches. Set to 900 for 15 minutes. This is optional.

Table 3 - Primary site intialization parameters

4.2.5.2 Standby Site Initialization Parameters

The following Initialization parameters need to be set on the standby site:

Parameter	Description
Remote_archive_enable	Enable remote archive log propagation.
Standby_file_management	Set to auto to ensure that database structure changes (adding tablespace/datafile) are automatically applied to the standby site.
FAL_SERVER	primary_DGTEST
FAL_CLIENT	local_DGTEST.macrotone.co.uk
Standby_archive_dest	/test/DGTEST/arc

Table 4 - Standby site initialization parameters

4.2.5.3 Sample Initialization File

Note: It is desirable to configure both sites (primary and standby) with values for both a standby and primary configuration. This ensures that in the event of a switchover everything is preconfigured.

```
log_archive_dest_1='LOCATION=/test/DGTEST/arc'
log_archive_format=DGTEST_%t_%s.arc
log_archive_start=true
log_archive_dest_2='SERVICE=standby_DGTEST.macrotone.co.uk LGWR ASYNC AFFIRM'
log_archive_dest_state_2=enable
REMOTE_ARCHIVE_ENABLE=TRUE
FAL_SERVER=primary_DGTEST.macrotone.co.uk
FAL_CLIENT=local_DGTEST.macrotone.co.uk
STANDBY_ARCHIVE_DEST=/test/DGTEST/arc
STANDBY_FILE_MANAGEMENT=auto
archive_lag_target=1800
```

4.2.6 Backup the Primary database.

The first step in the process is to take a consistent copy of the primary database; this can be achieved in one of 2 ways, either via a Cold Backup or a hot backup from a database, which is already running. Alternatively RMAN can be used to generate the backup – See Section 8.

4.2.6.1 Cold Backup

This is the simplest method to create a backup database. This is performed by:

1. Force a Redo log switch by performing:

```
sqlplus "/ as sysdba"
alter system archive log current;
alter system archive log current;
alter system archive log current;
```

alter system archive log current;

2. Cleanly shutdown the database. Bounce to be certain of cleanliness.

   sqlplus "/ as sysdba"

   shutdown immediate;

   startup;

   shutdown;

3. Using OS utilities copy all of the files which make up the database, including:

   Data files, control files, redo logs ($ORACLE_BASE/admin)

4. Go to Create Control file section.

4.2.6.2 Hot Backup

This is more complex but will allow the database to be backed up whilst still running.

Note : This can only be achieved if the database is running in archive log mode. Perform the following steps to achieve this:

1. Force a Redo log switch by :

   sqlplus '/ as sysdba';

   alter system archive log current;

   alter system archive log current;

   alter system archive log current;

   alter system archive log current;

2. Identify all of the datafiles making up the various tablespaces of the database.

   This can be achieved by looking at dba_data_files.

   Tablespace by tablespace perform the following:

   Put the tablespace into backup mode by issuing the command :

   alter tablespace xxx begin backup.

   Use OS utilities to copy the data files making up the tablespace.

   Take the tablespace out of backup mode by issuing the command:

   alter tablespace xxx end backup.

3. Use OS utilities to copy archive redo logs generated during the backup, and the current control file.

4. Create a standby control file.

4.2.7 Create a Standby Control File

The standby database requires a special control file to function. This control file is generated on the primary site by issuing the following commands:

sqlplus '/ as sysdba'

alter database create standby controlfile as 'filename';

Where the filename is a fully qualified operating system file name. Note that the database has to be in mount state for this command to work.

4.2.8 Create the Standby Site from the Backup Database

This process involves restoring the primary database backup to the standby site. Once this has been done then the following tasks also need to be performed:

1. Overwrite the control files on the standby site with the standby control file created above. Note: If there are 3 control files then each of them needs to be over written with the standby control file.
2. Place a copy of the initialization file in the appropriate admin directory usually

   $ORACLE_BASE/admin/DGTEST/pfile

4.2.9 Create a password file for the Standby Site

If using password files then a password file needs to be created for the standby database before it can be mounted.

This is achieved by issuing the command:

orapwd file=$ORACLE_HOME/dbs/orapwDGTEST password=password entries=5

Where password is the chosen password for the sysdba user.

4.2.10 Start the standby database

This involves starting and mounting the standby database ie.

sqlplus "/ as sysdba"

startup nomount;

alter database mount standby database;

4.2.11 Recovery of Standby database

If the standby database was created via the hot backup method then recover is required upon the standby database. The following statement needs to be run prior to placing the database in managed standby mode.

alter database recover automatic standby database;

The `AUTOMATIC` option automatically generates the name of the next archived redo log needed to continue the recovery operation.

After recovering the available logs, the Oracle database server prompts for the name of a log that does not exist. The reason is that the recovery process does not know about the logs archived to the standby site by the primary database.

For example, you might see:

```
ORA-00308: cannot open archived log '/oracle/standby/standby_logs/arcr_1_540.arc'
ORA-27037: unable to obtain file status
SVR4 Error: 2: No such file or directory
Additional information: 3
Specify log: {<RET>=suggested | filename | AUTO | CANCEL}
```

Cancel recovery after the Oracle database server has applied the available logs, by executing the following statement (or typing CTRL+C):

```
SQL> CANCEL
Media recovery cancelled.
```

Alternatively you may see a message as follows:

```
ORA-00279: change 3045922018453 generated at 11/29/2007 08:49:57 needed for thread 1
ORA-00289: suggestion : /oracle/standby_logs/arcr_0000016341
ORA-00280: change 3045922018453 for thread 1 is in sequence #16341
ORA-00278: log file '/oracle/standby_logs/arcr_0000016341' no longer needed for this recovery
ORA-16145: archival for thread# 1 sequence# 16341 in progress
```

In this situation issuing the following command may be appropriate.

SQL> alter database recover cancel;

The exact message that is seen is dependant upon the specific versions of the database in use.

The following error messages are acceptable after recovery cancellation and do not indicate a problem:

```
ORA-01547: warning: RECOVER succeeded but OPEN RESETLOGS would get error below
ORA-01194: file 1 needs more recovery to be consistent
ORA-01110: data file 1: 'some_filename'
ORA-01112: media recovery not started
```

Note that this will usually get as near up to date as possible before failing at the last log, which would not have been created at that time. It there are problems with the 'cancel' command it has been found useful to shutdown the database at this point and restart it again before putting it into managed recover mode. Attempts to put it into managed recovery mode immediately after issuing the above statement if the 'cancel' has not taken, generate a complaint about recovery is already active. Shutting down the database gets around this issue.

4.2.12 Enable Log Transport/Apply Services

Enabling Log Transport apply services are achieved by the following:

4.2.12.1 Primary Site:

Setting log_archive_dest_state_2=enable. This can be done either by enabling in the init.ora or by enabling via SQL plus.

4.2.12.2 Standby Site

The standby site needs to be placed into Managed recovery mode. This is achieved by issuing the statement:

alter database recover managed standby database disconnect;

4.2.13 Disable Log Transport/Apply Services

Disabling of Log Transport apply services is achieved by the following:

4.2.13.1 Primary Site:

Setting log_archive_dest_state_2=defer. This can be done either by enabling in the init.ora or by enabling via SQL plus.

4.2.13.2 Standby Site

The standby site needs to be placed into Managed recovery mode. This is achieved by issuing the statement:

alter database recover managed standby database cancel;

4.3 Temporary Files

By default temporary files associated with a temporary tablespace are not automatically created with a standby database.

Temporary files can be added to the temporary tablespace by issuing the command:

```
alter tablespace temp add tempfile '/test/SID/DGTEST/temp.dbf' size 1000M REUSE;
```

Note that the database has to be open before you can issue this command.

4.4 Forced Logging

The standby database is kept up to date by playing back transactions on the standby site, which have been recorded in the online redo logs. For efficiency purposes using the NOLOGGING option may have speeded up some database transactions. Usage of this feature in a Data Guard protected environment is undesirable.

From Oracle version 9.2, Oracle introduced a method to prevent these types of transaction occurring. This is known as forced logging. To enable forced logging issue the following command on the primary database:

```
alter database force logging;
```

5 Data Guard Options

5.1 Standby Redo Logs

Starting Oracle 9i you have to opportunity to add Standby Redo Log Groups to your Online Redo Log Groups. These Standby Redo Logs then store the information received from the Primary Database. In case of a Failover situation, you will have less data loss than without Standby Redo Logs.

Standby Redo Logs are only supported for the Physical Standby Database in Oracle 9i and as well for Logical Standby Databases in 10g. Standby Redo Logs are only used if you have the LGWR activated for archival to the Remote Standby Database.

The great Advantage of Standby Redo Logs is that every Entry written into the Online Redo Logs of the Primary Database is transferred to the Standby Site and written into the Standby Redo Logs at the same time; therefore, you reduce the probability of Data Loss on the Standby Database.

Starting with 10g it is possible to start Real-Time Apply with Physical and Logical Standby Databases. With Real-Time Apply, the Redo is applied to the Standby Database from the Standby Redo Log instead of waiting until an Archive Log is created. So Standby Redo Logs are required for Real-Time Apply.

5.1.1 How to create Standby Redo Logs

Standby Redo Logs are additional Redo Log Groups. If you were to query the V$LOGFILE view on the Standby Database, the output would typically look something like this:

```
SQL> select * from v$logfile;

 GROUP# STATUS  TYPE    MEMBER
 ------ ------- ------- ------------------------------------
      1          ONLINE C:\ORACLE\ORADATA\STANDBY\REDO01.LOG
      2          ONLINE C:\ORACLE\ORADATA\STANDBY\REDO02.LOG
      3          ONLINE C:\ORACLE\ORADATA\STANDBY\REDO03.LOG
```

Before you add the Standby Redo Logs to your Standby Database, verify the number of maximum Log file Groups and Log file Members. If you don't remember these values, you can look at the CREATE CONTROLFILE script. To get such a script, run the following command:

```
SQL> alter database backup controlfile to trace;
```

This will create a *.TRC-file in your UDUMP Directory. When you edit or view this newly created file, you will find a script to recreate the Controlfile. Notice the following relevant entries:

```
MAXLOGFILES   8
MAXLOGMEMBERS 3
```

In this example you can create a maximum of eight (8) Log file groups. Each group can contain a maximum of three (3) Members.

Now we can add Standby Redo Log files to the Standby Database (of course, the Standby Database must be in the MOUNT State):

```
SQL> alter database add standby logfile group 4
  2  ('C:\ORACLE\ORADATA\STANDBY\STBY04.LOG') SIZE 5M;
```

You can also add any further members to each group:

```
SQL> alter database add standby logfile member
  2  'C:\ORACLE\ORADATA\STANDBY\STBY14.LOG' to group 4;
```

Please keep in mind that if the RFS process is able to write into a Standby Redo Log, the file size of the Standby Redo Log MUST be equal to the current Online Redo Log of the Primary Database!!

5.1.2 Limitations to Standby Redo Logs

In a Oracle 9i/10g Data Guard Environment, the RFS Process on the Standby Database receives the Data from the Primary and writes it to Disk (either Standby Redo Logs or Archived Redo Logs).

If you consider using Standby Redo Logs, you must make certain that they are the same size as the Online Redo Logs. If you have different sizes of Online Redo Logs, you have to create corresponding Standby Redo Logs. The RFS process won't attach Standby Redo Logs if they are different from the Online Redo Log. It is recommended to have at least one more of Standby Redo Log Group as you have of Online Redo Log Groups per Thread and Size.

Standby Redo Logs are filled with the same information that is written to the Online Redo Logs of the Primary Database. Therefore, only the LGWR can provide this information to the Standby RFS process, so Standby Redo Logs will only benefit if you set LGWR as the Transmitter on the Primary Database in the LOG_ARCHIVE_DEST_n Initialization Parameter. Starting with 10.2.0, even the ARCH is able to write into Standby Redo Logs.

The RFS-Process always tries to allocate the next available Standby Redo Log, so it is possible that you encounter a Switch between only two Standby Redo Logs, although you created lots more of them. There's no rota defined here like in Online Redo Logs.

5.1.3 Differences in the Log Apply Services when using Standby Redo Logs

In case you do not have Standby Redo Logs, an Archived Redo Log is created by the RFS process and when it has completed, this Archived Redo Log is applied to the Standby Database by the MRP (Managed Recovery Process) or the Logical Apply in Oracle 10g when using Logical Standby. An open (not fully written) Archive Log file cannot be applied on the Standby Database and will not be used in a Failover situation. This causes a certain data loss.

If you have Standby Redo Logs, the RFS process will write into the Standby Redo Log as mentioned above and when a log switch occurs, the Archive Process of the Standby Database will archive this Standby Redo Log to an Archived Redo Log, while the MRP process applies the information to the Standby Database. In a Failover situation, you will also have access to the information already written in the Standby Redo Logs, so the information will not be lost.

Starting with Oracle 10g you have also the Option to use Real-Time Apply with Physical and Logical Standby Apply. When using Real-Time Apply we directly apply Redo Data from Standby Redo Logs. Real-Time Apply is also not able to apply Redo from partial filled Archive Logs if there are no Standby Redo Logs. So Standby Redo Logs are mandatory for Real-Time Apply.

5.2 File Name conversion

5.2.1 DB_FILE_NAME_CONVERT parameter

DB_FILE_NAME_CONVERT is useful for creating a duplicate database for recovery purposes. It converts the filename of a new data file on the primary database to a filename on the standby database. If you add a data file to the primary database, you must add a corresponding file to the

standby database. When the standby database is updated, this parameter converts the data file name on the primary database to the data file name on the standby database. The file on the standby database must exist and be writable, or the recovery process will halt with an error.

If you specify an odd number of strings (the last string has no corresponding replacement string), an error is signalled during database start-up. If the filename being converted matches more than one pattern in the pattern/replace string list, the first matched pattern takes effect. There is no limit on the number of pairs that you can specify in this parameter (other than the hard limit of the maximum length of multi-value parameters).

Set the value of this parameter to two strings. The first string is the pattern found in the data file names on the primary database. The second string is the pattern found in the data file names on the standby database.

You can also use `DB_FILE_NAME_CONVERT` to rename the data files in the clone control file when setting up a clone database during tablespace point-in-time recovery.

Parameter type String

Syntax

```
DB_FILE_NAME_CONVERT = [(]'string1' , 'string2' ,
'string3' , 'string4' , ...[)]
```

where:

`string1` is the pattern of the primary database filename
`string2` is the pattern of the standby database filename
`string3` is the pattern of the primary database filename
`string4` is the pattern of the standby database filename

One can use as many pairs of primary and standby replacement strings as required. It is possible to use single or double quotation marks. The parentheses are optional.

Following are example settings that are acceptable:

```
DB_FILE_NAME_CONVERT = ('/dbs/t1/','/dbs/t1/s_
','dbs/t2/ ','dbs/t2/s_')
```

Default value None

Parameter class: Static

5.2.2 Manual Renaming Files

Sometimes all of the primary datafiles and redo logs cannot be renamed in the standby control file by conversion parameters. For example, assume that your database has the following datafiles, which you want to rename as shown in the following table:

Primary Filename	Standby Filename
/oracle/dbs/df1.dbf	/standby/df1.dbf
/oracle/dbs/df2.dbf	/standby/df2.dbf
/data/df3.dbf	/standby/df3.dbf

You can set `DB_FILE_NAME_CONVERT` as follows to convert the filenames for the first two datafiles:

DB_FILE_NAME_CONVERT = '/oracle/dbs', '/standby'

Nevertheless, this parameter will not capture the renaming of `/data/df3.dbf`. You must rename this datafile manually in the standby database control file by issuing a SQL statement as follows:

```
SQL> ALTER DATABASE RENAME FILE '/data/df3.dbf' to '/standby/df3.dbf';
```

5.2.2.1 To rename a datafile manually

1. Start up and mount the standby database (if it is not already started) and then mount the database:

   SQL> STARTUP NOMOUNT PFILE=initSTANDBY1.ora;

   SQL> ALTER DATABASE MOUNT STANDBY DATABASE;

2. Issue an `ALTER DATABASE` statement for each datafile requiring renaming, where *`old_name`* is the old name of the datafile as recorded in the control file and *`new_name`* is the new name of the datafile that will be recorded in the standby control file:

   SQL> ALTER DATABASE RENAME FILE *'old_name'* TO *'new_name'*;

When you manually rename all of the datafiles that are not captured by the `DB_FILE_NAME_CONVERT` parameter, the standby database control file can correctly interpret the log stream during the recovery process.

5.2.3 LOG_FILE_NAME_CONVERT parameter

`LOG_FILE_NAME_CONVERT` converts the filename of a new log file on the primary database to the filename of a log file on the standby database. If one adds a log file to the primary database, one must add a corresponding file to the standby database.

If ones specifies an odd number of strings (the last string has no corresponding replacement string), an error is signalled during database start-up. If the filename being converted matches more than one pattern in the pattern/replace string list, the first matched pattern takes effect. There is no limit on the number of pairs that you can specify in this parameter (other than the hard limit of the maximum length of multi-value parameters).

When the standby database is updated, this parameter converts the log file name on the primary database to the log file name on the standby database. The file must exist on the standby database and must be writable or the recovery process will halt with an error.

The first string is the pattern found in the log file names on the primary database. The second string is the pattern found in the log file names on the standby database.

One should also use `LOG_FILE_NAME_CONVERT` to rename the log files in the clone control file when setting up the clone database during tablespace point-in-time recovery.

Parameter type: String

Syntax

```
LOG_FILE_NAME_CONVERT = [()'string1' , 'string2' ,
'string3' , 'string4' , ...[)]
```

where:

`string1` is the pattern of the primary database filename
`string2` is the pattern of the standby database filename
`string3` is the pattern of the primary database filename
`string4` is the pattern of the standby database filename

One can use as many pairs of primary and standby replacement strings as required. It is possible to use single or double quotation marks. The parentheses are optional.

Following is an example of settings that are acceptable:

```
LOG_FILE_NAME_CONVERT=('/dbs/t1/','/dbs/t1/s_
','dbs/t2/ ','dbs/t2/s_')
```

Default value:	None
Parameter class	Static
Range of values	Character strings

6 Enterprise Manager/Grid Control

The easiest way to manage and monitor a Data Guard environment is to use the Data Guard manager (part of Enterprise Manager/Grid Control).

The Data Guard manager can be used to create the Data Guard database in the first instance, although it is often preferable to create and test the Data Guard database first and then discover it using the Data Guard Manager Wizard.

This section of the report details the steps required to discover the Data Guard environment created above:

Note: All of the information detailed below relates to Grid control, although the process is similar when using Enterprise Manager.

6.1 Accessing Data Guard Manager

Access the Data Guard Web pages through the Oracle Enterprise Manager Grid Control using the following steps:

1. Click the Targets tab to go to the Targets page.
2. Click Databases to go to the Databases page.
3. On the Databases page, you see a listing of all discovered databases. In this scenario, the primary database, North_Sales, has already been discovered. Click the North_Sales database to go to the primary database home page.
4. Click Maintenance.
5. In the High Availability section under Data Guard, click Setup and Manage and log in.

Note:

If you log in as a user with SYSDBA privileges, you will have access to all Data Guard functionality, including all monitoring and management features. If you log in as a non-SYSDBA user, you will have access to monitoring functions only; features such as standby creation, switchover, and failover will not be available.

If the primary database is not already in a broker configuration, clicking Setup and Manage go to the page shown in below. This screenshot shows information presented on the All Targets tab, containing graphs for all targets discovered for the cluster.

From the list of managed targets, you can quickly assess the availability of the targets and click any graph to drill down for more information.

Figure 2 - 10G EM Database display

6.2 Adding an existing standby database

Creating a broker configuration is the first thing you must do before you can manage and monitor the databases. Enterprise Manager provides the Add Standby Database wizard to create a broker configuration that includes a primary database and one or more standby databases. This scenario shows how to use the Add Standby Database wizard later to add a logical standby database to a broker configuration that already has one physical standby database (DR_Sales). To start the Add Standby Database wizard, click Add Standby Database on the Data Guard Overview page.

Manage an existing standby database with Data Guard broker.

You will need to connect to the primary database using SYSDBA credentials, if you are not yet connected.

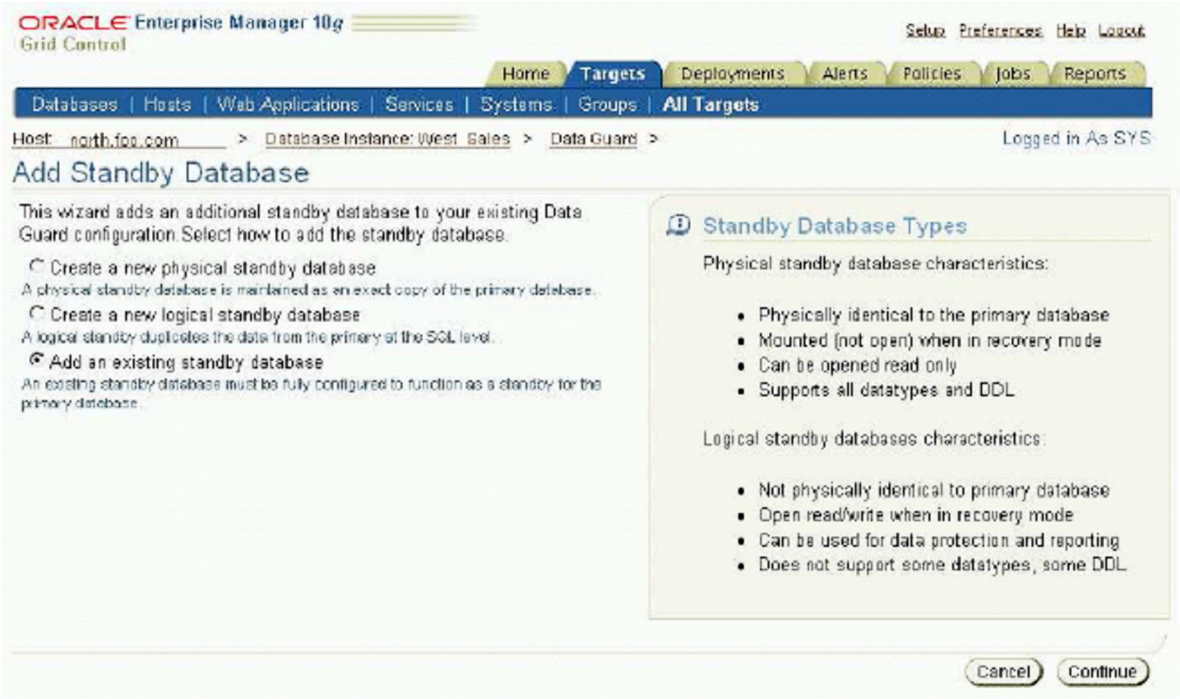

Figure 3 - 10g EM Standby Addition

6.2.1 Select an existing standby database.

In this step, select the standby database you want to add to the configuration. All discovered databases in your environment will be shown in the list. In the example shown in (Step 1 of 3), one of the databases in the display is your standby database.

You can click Cancel at any time to terminate the current process and begin again at the introductory page of the Add Standby Database wizard.

If you wish to continue, click Next.

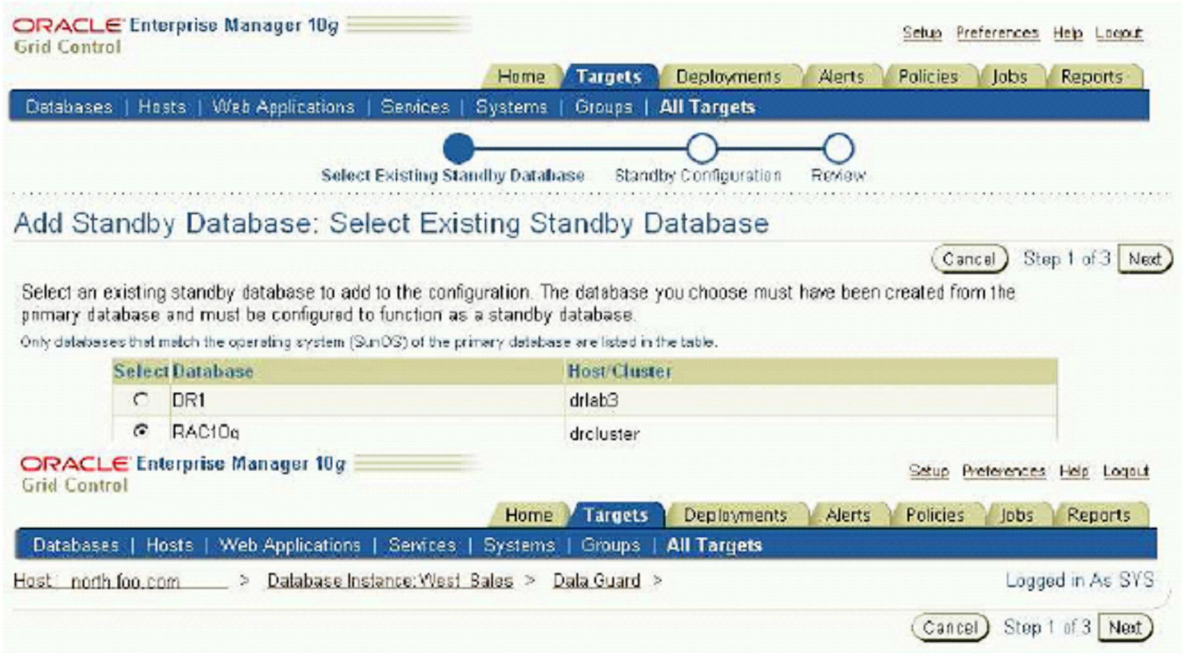

Figure 4 - 10G EM adding existing Standby DB

6.2.2 Set the standby archive location setting.

You can optionally change the Standby Archive Location setting of the existing standby cluster database, as shown in (Step 2 of 3).

If you wish to continue, click Next.

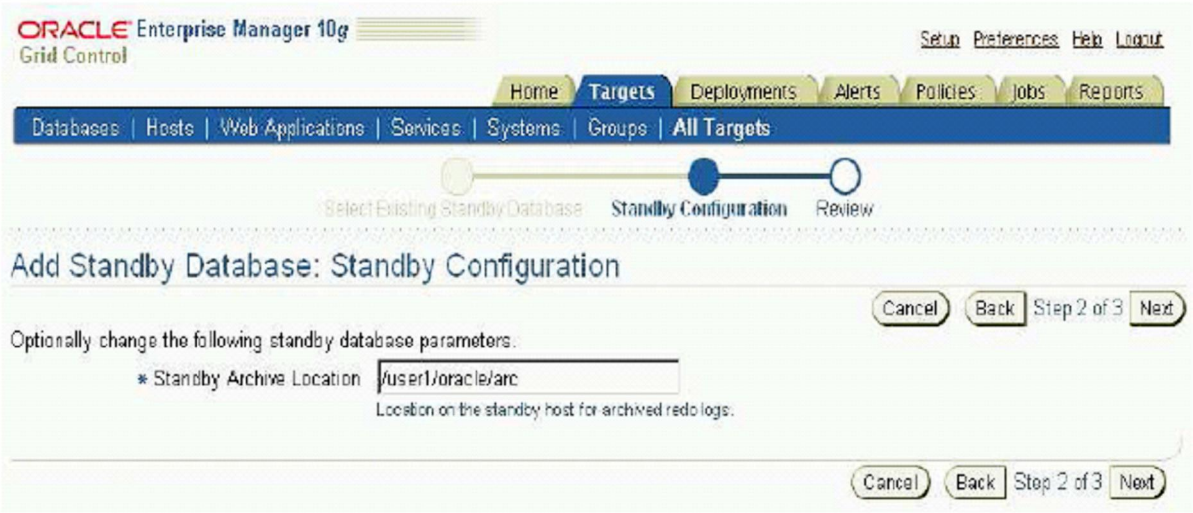

Figure 5 - 10G EM Setting archive location

6.2.3 Step 3 Review the configuration settings.

Review the data for the configuration and standby database, as shown in (Step 3 of 3). You can click Cancel to terminate the current process and begin again at the introductory page of the Add Standby Database wizard.

If you wish to complete the operation, click Finish

Figure 6 - 10G EM Configuration review

7 Failover

7.1 Forced Failover – No Redo Logs

If the primary database is not available because of disaster then a forced failover will be required to the standby site. If possible then the online redo logs should be recovered from the primary site. If this is not the case then the following steps must be undertaken. This is destructive; once the standby site has become activated then the primary and/or any other standby sites will require rebuilding.

Activating the standby database in this scenario is accomplished by typing the following command having applied all outstanding log changes (see above):

```
alter database recover managed standby database finish;

alter database commit to switchover to primary;

alter database open resetlogs;
```

7.2 Forced Failover – Redo logs

If the primary database is not available because of disaster then a forced failover will be required to the standby site. If possible then the online redo logs should be recovered from the primary site. This is destructive; once the standby site has become activated then the primary and/or any other standby sites will require rebuilding.

If the online redo logs are available then the following steps will need to be performed:

1. Shutdown the primary and standby databases with the immediate option.
2. Restore the online redo logs to the standby site.
3. Startup and mount the standby database

   ```
   sqlplus "/ as sysdba"

   startup nomount;

   alter database mount standby database;
   ```

4. Create a script to create the database control files.

   ```
   alter database backup controlfile to trace;
   ```

   This will create a file in $ORACLE_BASE/admin/$ORACLE_SID/udump.
5. Extract from this file the statement to create the control file and place it into a sql script file;
6. Shutdown the standby database;

   ```
   shutdown immediate;
   ```

7. Run the sql script to create the control file.
8. Restart the database.

   ```
   shutdown immediate;

   startup;
   ```

7.3 Graceful failover

When failing over to the standby site because of a planned activity such as maintenance on the primary server then the fail over process is non destructive. The process can be performed in reverse when the primary site becomes available, without having to rebuild either database.

Activating the standby database in this scenario is accomplished by typing the following command having applied all outstanding log changes (see above):

On the primary database issue the following command:

```
alter database commit to switchover to standby;
```

On the standby database issue the following command;

```
alter database commit to switchover to primary;
```

shutdown and restart the database.

Note: If the primary site is going to be unavailable ensure that log_archive_dest_state_2 is set to defer during the downtime.

8 Using RMAN to Create the Standby Database

An alternative method to create the standby database would be to use Oracle Recovery Manager. The steps required to do this are outlined below.

8.1 Assumptions:

The process can be more complex and flexible however the procedure below makes the following assumptions:

- ÅStandby Database structure is the same as the Primary database.
- ÅA Recovery catalog exists.

8.2 RMAN Backup

There are two methods of creating an rman backup which is suitable for the creation of a standby database.

8.2.1 New Backup

If the database has never been backed up before using RMAN then a backup needs to be made:

Ie.

```
rman target / catalog rman/rman@rmandb
register database;

Run {
        allocate channel d1 device type disk format ∂backup/%Uô
        allocate channel d2 device type disk format ∂backup/%Uô
        backup database include current controlfile for standby plus archivelog;
}
```

In RAC implementations where it may not be possible to see all of the archive logs issue the command:

```
Run {
        allocate channel d1 device type disk format ∂backup/%Uô
        allocate channel d2 device type disk format ∂backup/%Uô
        sql ñalter system archive log currentò
        backup database include current controlfile for standby;
        sql ñalter system archive log currentò
}
```

8.2.2 Existing Backup

If the database has previously been backed up then a standby control file needs to be created, this is done by:

```
rman target / catalog rman/rman@rmandb
register database;
run {
        allocate channel d1 device type disk format ∂backup/%Uô
        allocate channel d2 device type disk format ∂backup/%Uô
        backup current controlfile for standby;
        sql ∂alter system archive log current;ô
        backup archive log all not backed up 1 times;
}
```

8.3 Creating the Standby Database

8.3.1 Pre-requisites

Before building the standby database, ensure that:

- Oracle executables are installed on the standby machine.
- Database directory structure exists on the standby machine.
- Ensure that the listener is started on the standby machine.
- Ensure that the standby database is statically registered with the standby listener;
- Ensure that a tnsnames entry exists on the primary, which points to the standby database;
- Ensure that the primary initialization file (init.ora) is available on the standby host.
- Ensure that in the initialization file remote_password_file is set to none.
- Ensure that the standby database has a password file created.
- Ensure that RMAN backup pieces are available to the standby host.

8.3.2 Procedure

The following example assumes a recovery catalogue is being used. This is not a mandatory requirement.

1. Startup the standby database instance:

```
startup nomount pfile=init.ora
```

2. Connect to RMAN

```
rman target / catalog rman/rman@rmandb auxillary sys/syspwd@standby
```

3. Create the standby database:

```
run {
      allocate channel d1 device type disk;
      allocate auxiliary channel a1 device type disk;
      duplicate target database for standby nofilenamecheck;
}
```

The Standby database will now be created.

9 Changing Standby to Read ONLY mode.

9.1 Archivelogs from Primary in READ ONLY mode

Even though the Standby database may be in read only mode, archived log files from the Primary database will continue to be sent to the Standby database. However, archived log files received by the Standby database in read only mode will not be applied to the Standby database.

This means that the Standby data base will only be current with the Primary database up to the time the Standby database was put into read only mode.

When the Standby database is changed from read only mode to Managed Recover Mode, all of the accumulated archived log files from the Primary database will be applied and the Standby data base will be "current" with the Primary database.

9.2 Opening the database in READ ONLY mode

At the UNIX prompt connect to the standby database using SQLPLUS:

```
SQL> startup nomount

ORACLE instance started.

SQL> alter database mount standby database;

Statement processed.

SQL> select instance_name,status from v$instance;

INSTANCE_NAME    STATUS
---------------- -------
std              MOUNTED
1 row selected.

SQL> select name, open_mode, controlfile_type from v$database;

NAME      OPEN_MODE  CONTROL
--------- ---------- -------
PRIM    MOUNTED    STANDBY

SQL> alter database open read only;

Statement processed.

SQL> select instance_name, status from v$instance;

INSTANCE_NAME    STATUS
---------------- -------
std              OPEN
1 row selected.

SQL> select name, open_mode, controlfile_type from v$database;

NAME      OPEN_MODE  CONTROL
--------- ---------- -------
PRIM      READ ONLY  STANDBY
1 row selected.
```

9.2.1 Error in alert log.

You will probably receive messages in the alert log such as: 'Executing transaction without active Undo Tablespace'. This is normal and the expected behaviour since no Undo segments are onlined during a database which is open in READ ONLY mode.

9.3 From Read Only to Managed Recovery Mode

To place a read only database back into managed recovery mode it should only be necessary to issue a single command:

```
ALTER DATABASE RECOVER MANAGED STANDBY DATABASE DISCONNECT;
```

The system should then close down all active sessions and enter recovery mode. The author has experienced a few problems with this. The most recent was that the sessions disconnected and then the whole database seemed to be effectively idle. Whether this was due to the size of the database concerned or whether the author was just impatient is not known. The recourse was to abort the instance and restart from a new mount prior to entering managed recover mode.

There are a number of reported problems with the closing of sessions in this situation, but these are very likely very database version dependant.

10 Usage for UAT environments

10.1 Preparation

Open two UNIX sessions/windows – one on the primary database system and one on the standby database system. Login to the oracle UNIX account on both servers.

Tail the Oracle database alert log on the primary and standby databases using tail –f <alert log name> to monitor for errors/performance issues on either the production or the standby database or the production/standby server.

To locate the alert logs, type the command bdump (an alias set up in the Oracle account) which sets the current directory to the alert log directory. The alert log for both databases will be identical and are called alert_*SID*.log.

tail –f alert_*SID*.log.

Open a 2nd UNIX session/window on the primary system and again log into the Oracle UNIX account. Now login to the primary database as follows:-

sqlplus '/ as sysdba'

Now execute the following commands within SQLPLUS, to halt replication from the primary database to the standby database.

alter system set dg_broker_start =false scope=both; (Can be ignored if not using the Data Guard broker.)

alter system set log_archive_dest_state_3=defer scope=both;

Now type exit to logout of sqlplus.

(Note: If you don't stop the dg broker on production if it is running, then as soon as the standby db comes back up it will automatically switch the log_archive_dest_state_x back to enable and start pushing logs)

Change to the archive log directory on the production system and make a note of the latest/most recently created archive log file name on the primary database. The archive logs are created with ascending numbers, so when we come to re-enable replication (the pushing of archive logs from production to standby) later on, we can see how many archive logs have been created while replication was disabled.

Monitor the alert logs on the primary database and standby database, in the first two windows/sessions we opened, to check there are no errors/performance issues on either the production or the standby database or the production/standby server.

10.2 Backup standby database

From the UNIX window logged onto the standby system as oracle, logon to the standby database as follows :-

sqlplus '/ as sysdba'

Make a copy of the SPFILE on the standby system

create pfile ='init*SID*.dt' from spfile; (this creates backup file init*SID*.dt in the $ORACLE_HOME/dbs directory)

alter database backup controlfile to trace;

and save trace file (which is created in udump) to pre-backup.ctl

Now shutdown the standby database.

```
shutdown immediate;
```

Now type exit to logout of sqlplus.

Now shutdown the listener on the standby system, as follows :-

```
lsnrctl
set password <password>          (If a password is used.)
stop
exit
```

Backup all of the database files and directories containing the database datafiles, controlfiles, redo logs and config files etc.)

If there is sufficient file system space it might be possible to just copy the files to other directories, but for a large database this may be prohibitive.

10.3 Fully open standby database

Under its normal configuration, the standby database is not fully open/available to users, but is in a special 'recovery mode' where it is sits there waiting for updates from the production database in Reading and then applies them.

For the purpose of UAT testing we need to act as if there has been a disaster on the primary database, in which case, the standby database needs to be switched out of standby/recovery mode and opened fully for user access. From the moment we open the standby database fully though, it is no longer useable as a standby database in its current form (although we will later restore the database from our pre-backup, so that it is restored back to its useable standby state). So, for all intents and purposes from this moment onwards the current database as it currently exists serves no other purpose than to act as our UAT testing database.

Login to the Oracle Unix account on the standby system and logon to the standby database as follows :-

```
sqlplus '/ as sysdba'
```

Now bring the database back up again, and disable the standby/recovery mode.

```
startup nomount;
alter database mount standby database;
```

For a UAT testing environment one must ensure that there is no possibility of any archive logs being shipped and applied to our primary database. Therefore ensure that the log destination on the remote is not active.

```
alter system set log_archive_dest_x='' scope=both;
alter database activate physical standby database skip standby logfile;
```

At this point, you may well see a raft of erroneous looking messages in the alert_*SID*.log file, one for each of the redo logs (these messages can be safely ignored)

Now shutdown the database and bring it back up in fully-open mode, so it's available to users for UAT testing.

shutdown immediate;

startup

select * from dba_temp_files; and select * from v$tempfile;

To check if temp files have been added/created for the temp tablespace, if not

alter tablespace temp add tempfile '/data/filesys/oradata/*SID*/temp01.dbf' size 3000M;

Size tempfiles as appropriate for the system. If tempfile already exists add REUSE keyword to end of the previous command.

Type exit to logout of sqlplus.

We are now ready for users to do the full UAT system testingf.

10.4 User system testing

Ensure all users performing the testing are using the 'UAT' test system and not acting upon the 'real' live system.

Once the UAT system tests are completed, then the standby system needs to be re-established.

10.5 Post backups

Once UAT testing is complete, we can then shutdown the UAT testing database and if required take a full a full cold backup of the database (this post backup is optional and may not be required).

Login to the Oracle Unix account on the standby system and logon to the standby UAT testing database as follows:-

sqlplus '/ as sysdba'

Firstly, take a backup of the controlfile to trace

alter database backup controlfile to trace;

Rename file (in udump) to post_backup.ctl.

Now shutdown the database.

shutdown immediate;

Type exit to logout of sqlplus.

Perform any backup as required of the completed test environment.

10.6 Restore pre-backups

We now need to restore the standby database to its pre-backup state (i.e restore everything to the state it was the last time the production/primary database was talking to it). So, we need to restore the pre-backup datafiles etc, and then re-start the pre-backup database.

Login as Oracle on the standby system server and restore the pre-backup datafiles back to standby system.

NOTE:

If there has been a large period of time that has elapsed between starting UAT testing and completing the testing then a log of archive logs would have been created on the primary database in the intervening time. If may be quicker to copy all the archive logs over to the

standby system and apply then prior to restarting the log shipping from the primary database itself. This would involve starting the standby database and applying all the outstanding logs prior to re enabling the primary archiving to the standby database.

Some system also have a technique of compressing the archive logs once they are generated on the primary to save disk storage space on the primary. In this situation it would be necessary to ensure that the archive logs are uncompressed, and in their original location otherwise the primary database would be unable to locate the logs and ship them to the standby system.

10.7 Restoring archivelogs using RMAN

Using the RMAN utility it is easy to restore individual archive logs to a system. It is often necessary to restore the logs to a new location and this can be performed using a script as shown below:

```
RUN
{
  # Set a new location for logs 1 through 100.
  SET ARCHIVELOG DESTINATION TO '/fs1/tmp';
  RESTORE ARCHIVELOG FROM SEQUENCE 4242 UNTIL SEQUENCE 4250;
  # Set a new location for logs 101 through 200.
  SET ARCHIVELOG DESTINATION TO '/fs2/tmp';
  RESTORE ARCHIVELOG FROM SEQUENCE 4251 UNTIL SEQUENCE 4260;
}
```

Note that the file location that are restored to, are specified in the above script. It is also noted that the next 'normal' backup of the database and archivelogs also backs up the restore logs and it the script is also intended to remove backed up archivelogs they are also removed from the restored location. This means that once restored the logs need to be moved to the standby location (if not placed there directly) prior to the next backup, otherwise they would need restoring again!.

10.8 Restart replication from primary db

To completely restore the production environment, we now need to restart replication from the primary database to the restored standby database.

Open two UNIX sessions/windows – one on the primary system and one on the standby system. Login to the oracle UNIX account on both servers. In both sessions run bdump alias to switch to the alert log directory, and then run tail –f alert_*SID*.log to tail output to both the alert logs.

In the Unix session on the primary system and logon to the primary database as follows :-

```
sqlplus '/ as sysdba'
```

Now execute the following commands within sqlplus, to restart the dg broker if it is being used.

If running the Data Guard Broker.

```
alter system set dg_broker_start=true scope=both;
```

If not just enable archive log shipping

```
alter system set log_archive_dest_state_x=enable scope=both;
```

Login to the Oracle Unix account on the standby system and logon to the restored standby database as follows :-

```
sqlplus '/ as sysdba'
```

Now startup the standby database.

```
startup nomount;
```

Now type exit to logout of sqlplus.

Now startup the listener on the standby system.

```
lsnrctl
set password <password>          If being used.
start
exit
```

(Simply switching on the Data Guard broker on the primary system, and startup nomount-ing the standby database and starting up the listener on the standby system, should be enough to restart the replication process again. The primary database should automatically switch log_archive_dest_state_x back to enable, without any intervention).

After a minute or so, you should see output that indicates that the intermediate archive log files (created on production while replication was disabled) are being created and pushed across to the standby database.

In archive directory on the standby system one should start to see the new archive logs begin to appear.

The UAT tests should now be complete and the database is back in its original state, receiving logs and applying them, from the primary database.

11 Database patching/upgrading

There are a few important steps to be considered when upgrading or patching databases which are part of a primary/standby configuration. It is important to read the patch (or upgrade) instructions carefully before proceeding. Remember that there is virtually twice the amount of work to be performed as if it were only a single stand alone database. The following is a brief overview of the important steps required to apply a patch set to the system(s).

11.1 Patch Set Apply Procedure

1. Log in to the oracle account on both the primary and standby hosts and make sure the environment is set to the correct ORACLE_HOME and ORACLE_SID.

2. On both the primary and standby host uncompress and untar the downloaded patch set / interim patch file into a new directory.

3. Shut down the existing Oracle Server instance on the primary host with normal or immediate priority. Stop all listeners, agents and other processes running against the ORACLE_HOME. If using Real Application Clusters perform this step on all nodes.

```
SQL> shutdown immediate

% agentctl stop

% lsnrctl stop
```

4. Cancel managed recovery on the standby database.

```
SQL> recover managed standby database cancel;
```

5. Shutdown the standby instance on the standby host. Stop all listeners, agents and other processes running against the ORACLE_HOME. If using Real Application Clusters perform this step on all nodes.

```
SQL> shutdown immediate

% agentctl stop

% lsnrctl stop
```

6. Run the Installer, or the the 'opatch' utility and install the patchset on both primary and standby host.

```
% .opatch apply

OR

%  runInstaller
```

If this is an interim patch, run opatch per the patch README.

If using Real Application Clusters, be sure the install has propagated to the other nodes if using private ORACLE_HOMEs. Please see the Patch Set readme for specific instructions.

7. Once the patchset/patch has been installed on on all hosts / nodes startup the standby listener standby host.

```
% lsnrctl start
```

8. Startup nomount the standby database.

```
% sqlplus "/ as sysdba"
```

```
SQL> startup nomount
```

9. Mount the standby database.

```
SQL> alter database mount standby database;
```

10. Place the standby database in managed recovery mode.

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

11. Startup the primary instance on the primary host.

```
% sqlplus "/ as sysdba"

SQL> startup migrate
```

12. Ensure that remote archiving to the standby database is functioning correctly by switching logfiles on the primary and verifying that v$archive_dest.status is valid. If you are not performing remote archiving make note of the current archive log sequence.

```
SQL> alter system archive log current;

SQL> select dest_id, status from v$archive_dest;
```

13. On the primary instance run the following script:

```
SQL> @?/rdbms/admin/catpatch.sql
```

For the interim patch, run any scripts as outlined in the README.

14. Once the catpatch.sql script / patch SQL scripts completes make note of the current log sequence and issue the following command:

```
SQL> alter system archive log current;
```

15. Verify the standby database has been recovered to the log sequence from step 12.

```
SQL> select max(sequence#) from v$log_history;
```

16. On the primary instance run the following command:

```
SQL> alter system disable restricted session;
```

17. Complete the remainder of the "Post Install Actions" from the Patch Set readme on the primary host. Please note that it is not necessary to shutdown the standby in conjunction with the primary during the "Post Install Actions".

18. Once all "Post Install Actions" have been completed verify the standby database has been recovered to the last archive log produced by the primary.

On the primary:

```
SQL> select max(sequence#) from v$archived_log;
```

On the standby:

```
SQL> select max(sequence#) from v$log_history;
```

12 Monitoring Standby Database

A certain amount of monitoring is required to ensure that the standby database is kept up to date with the primary. In addition to the usual checking of the database alert logs (both primary and standby) the following are a few scripts that might prove useful.

12.1 Network Failure Preventing the Archiving of Logs to the Standby Site

If you maintain a Data Guard or standby environment, and the network goes down, the primary database might continue to archive to disk but be unable to archive to the standby site. In this situation, archived logs accumulate as usual on the primary site, but the standby instance is unaware of them.

To prevent this problem, you can specify that the standby destination have mandatory status. If the archiving destination is mandatory, then the primary database will not archive any logs until it is able to archive to the standby site. For example, you can set the following in the primary initialization parameter file to make `standby1` a mandatory archiving destination:

```
LOG_ARCHIVE_DEST_2 = 'SERVICE=standby1 MANDATORY'
```

One consequence of this configuration is that unless the network problem is fixed, the primary database eventually stalls because it cannot switch into an unarchived online redo log. This problem is exacerbated if you maintain only two online redo logs in your primary database. For these reasons it is often recommended to not make the standby log destination mandatory.

12.2 Archived logs not applied on a physical standby database

To determine if there is an archive gap, query the `V$ARCHIVED_LOG` and `V$LOG` views. If an archive gap exists, the output of the query specifies the thread number and log sequence number of all logs in the archive gap. If there is *no* archive gap for a given thread, the query returns no rows.

12.2.1 To identify the logs in the archive gap

Query the `V$ARCHIVED_LOG` and `V$LOG` views on the standby database. For example, the following query shows that there is a difference in the `RECD` and `SENT` sequence numbers for the destination specified by `DEST_ID=2`, indicating that there is a gap:

```
SQL> SELECT MAX(R.SEQUENCE#) LAST_SEQ_RECD, MAX(L.SEQUENCE#) LAST_SEQ_SENT FROM
  2> V$ARCHIVED_LOG R, V$LOG L WHERE
  3> R.DEST_ID=2 AND L.ARCHIVED='YES';

LAST_SEQ_RECD LAST_SEQ_SENT
------------- -------------
            7            10
```

Use the following query to determine the names of the archived redo logs on the local system that must be copied to the standby system that has the gap:

```
SQL> SELECT NAME FROM V$ARCHIVED_LOG WHERE THREAD#=1 AND DEST_ID=1 AND
  2> SEQUENCE# BETWEEN 7 AND 10;

NAME
--------------------------------------------------------------------------------
/primary/thread1_dest/arcr_1_7.arc
/primary/thread1_dest/arcr_1_8.arc
/primary/thread1_dest/arcr_1_9.arc
/primary/thread1_dest/arcr_1_10.arc
```

1. Connect to the standby and get the sequence number of the last applied archive log. Call it laseq:

```
select max(sequence#) from v$archived_log where applied='YES';
```

2. Get the sequence number of the last complete archive log on the standby. This is the last log the standby can apply without receiving additional archive logs from the primary. Call it lrseq:

```
SELECT min(sequence#) FROM v$archived_log WHERE ( (sequence#+1) NOT IN

(SELECT sequence# FROM v$archived_log) ) AND (sequence# > laseq)
```

4. The difference between the two values obtained above (lrseq - laseq) is the number of archived logs that have not been applied on the standby, but could be applied if the primary host becomes unavailable. If a failover occurs, this number indicates the number of archive logs that need to be applied before activating the standby.

12.3 Manually Transmitting the Logs in the Archive Gap to the Standby Site

After you have obtained the log sequence numbers of the logs in the archive gap, you can obtain their filenames by querying the `V$ARCHIVED_LOG` view on the primary site. The archived log filenames on the standby site are generated by the `STANDBY_ARCHIVE_DEST` and `LOG_ARCHIVE_FORMAT` parameters in the standby initialization parameter file.

If the standby database is on the same site as the primary database, or the standby database is on a remote site with a different directory structure than the primary database, the filenames for the logs on the standby site cannot be the same as the filenames of the logs archived by the primary database. Before transmitting the archived logs to the standby site, determine the correct filenames for the logs at the standby site.

12.3.1 To copy logs in an archive gap to the standby site

1. Review the list of archive gap logs that you obtained earlier. For example, assume you have the following archive gap:

```
THREAD#    LOW_SEQUENCE#   HIGH_SEQUENCE#
---------- -------------   --------------
       1             460              463
       2             202              204
       3             100              100
```

   If a thread appears in the view, then it contains an archive gap. You need to copy logs from threads 1, 2, and 3.

2. Determine the filenames of the logs in the archive gap that were archived by the primary database. After connecting to the primary database, issue a SQL query to obtain the name of a log in each thread. For example, use the following SQL statement to obtain filenames of logs for thread 1:

```
SQL> SELECT NAME FROM V$ARCHIVED_LOG WHERE THREAD#=1 AND DEST_ID=1
  2> AND SEQUENCE# > 459 AND SEQUENCE# < 464;

NAME
--------------------------------------------------------------------------
/primary/thread1_dest/arcr_1_460.arc
/primary/thread1_dest/arcr_1_461.arc
/primary/thread1_dest/arcr_1_462.arc
/primary/thread1_dest/arcr_1_463.arc
4 rows selected
```

3. On the standby site, review the settings for `STANDBY_ARCHIVE_DEST` and `LOG_ARCHIVE_FORMAT` in the standby initialization parameter file. For example, you discover the following:

   STANDBY_ARCHIVE_DEST = /standby/arc_dest/

   LOG_ARCHIVE_FORMAT = log_%t_%s.arc

   These parameter settings determine the filenames of the archived redo logs at the standby site.

4. On the primary site, copy the archive gap logs from the primary site to the standby site, renaming them according to values for `STANDBY_ARCHIVE_DEST` and `LOG_ARCHIVE_FORMAT`. For example, enter the following copy commands to copy the archive gap logs required by thread 1:

   ```
   % cp /primary/thread1_dest/arcr_1_460.arc /standby/arc_dest/log_1_460.arc
   % cp /primary/thread1_dest/arcr_1_461.arc /standby/arc_dest/log_1_461.arc
   % cp /primary/thread1_dest/arcr_1_462.arc /standby/arc_dest/log_1_462.arc
   % cp /primary/thread1_dest/arcr_1_463.arc /standby/arc_dest/log_1_463.arc
   ```

   Or use the rcp command if the standby location is on a remote host.

   Perform similar copy commands to copy archive gap logs for threads 2 and 3.

5. On the standby site, if the `LOG_ARCHIVE_DEST` and `STANDBY_ARCHIVE_DEST` parameter values are *not* the same, then copy the archive gap logs from the `STANDBY_ARCHIVE_DEST` directory to the `LOG_ARCHIVE_DEST` directory. If these parameter values *are* the same, then you do not need to perform this step.

   For example, assume the following standby initialization parameter settings:

   STANDBY_ARCHIVE_DEST = /standby/arc_dest/

   LOG_ARCHIVE_DEST = /log_dest/

   Because the parameter values are different, copy the archived logs to the `LOG_ARCHIVE_DEST` location:

   % cp /standby/arc_dest/* /log_dest/

   If the standby location is on another machine which is usually the situation use the rcp command instead.

   % rcp /standby/arc_dest* remote:/log_dest/*

   When you initiate manual recovery, the Oracle database server looks at the `LOG_ARCHIVE_DEST` value to determine the location of the logs.

   Now that all required logs are in the `STANDBY_ARCHIVE_DEST` directory, you can proceed to section 4.2.11 to apply the archive gap logs to the standby database.

12.4 Potential data loss window for a physical standby database

1. Connect to the standby and get the sequence number of the last applied log. Call it laseq:

```
select max(sequence#) from v$archived_log where applied='YES';
```

2. Get the sequence number of the last complete archive log on the standby. This is the last log the standby can apply without receiving additional archive logs from the primary. Call it lrseq:

```
SELECT min(sequence#) FROM v$archived_log WHERE ( (sequence#+1) NOT IN

(SELECT sequence# FROM v$archived_log) ) AND (sequence# > laseq)
```

3. Connect to the primary database and obtain the sequence number of the current online log, call it curseq.

```
select sequence# from v$log where status='CURRENT';
```

4. The difference between the current online log sequence number and the last completed sequence no of the standby (curseq - lrseq) is the number of archive logs that the standby database would not be able to recover should the primary host become unavailable.

12.5 Determining the amount of redo sent to the remote destination

Run the following query on the standby to determine the number of blocks received from the primary and written to the standby redo logs:

```
select to_char(sysdate,'dd-MON-yy hh:mi:ss') TIMESTAMP,

THREAD#, SEQUENCE#, BLOCK#, BLOCKS, DELAY_MINS

from v$managed_standby

where process='RFS' and status='WRITING';
```

NOTES:

1) The above procedures will work both for 9iR1 and 9iR2.

2) The above procedures work for a non-RAC primary and non-RAC standby setup. If RAC database is involved, the above procedure needs to be run against each thread of the redo.

3) For potential data loss computation, there is an exception to the above calculation: If the primary is using LGWR to ship redo to standby redo logs and curseq-lrseq=1, the redo in the primary's current online log may already be shipped to the standby redo logs on the standby, in which case there is no data loss for the standby. In this case, query v$archive_dest_status to get the PROTECTION_MODE column of the corresponding standby. If the value is RESYNCHRONIZATION, that means there is data loss. Otherwise, consider no data loss for the standby.

13 Considerations for changing primary

The following considerations may or may not need to be considered for each configuration.

13.1 Temporary Tablespace

There will be a need to define a temporary tablespace data file for the standby database prior to it being switched to a primary. To do this the database has to be in an open mode. Until such time as it has been opened it will not be possible to add a suitable data file.

13.2 Redo Logs definitions and layout

It is often common unless the file systems are identical upon the primary and the standby systems that some form of mapping will have been required when placing data files upon the standby. In a recent configuration the source system had over 30 disks whilst the standby was equipped with only 5 very large file systems. This resulted in the data files on the standby not necessarily being best placed for the new system. Upon switch over some form of reconfiguration might well be in order.

13.3 Directory definitions

The primary database may well have a number of defined Oracle directories for the application to read or write to operating system files. If the file system layouts are different between the primary and the standby hosts then immediately after opening the standby database there will be a need to redefine the Oracle directories to reflect the new (standby) file structure.

13.4 System file directory mappings.

There may be difficulties in defining the appropriate configuration parameters for db_file_name_convert and log_file_name_convert when the file structures of the primary and the standby systems are radically different. One example is where the primary has a lot of small file systems, but the standby has fewer large file systems. It might be possible to use symbolic links to get around the issue, but the author has not tried this. Other wise the ability to switch between primary and standby at will be will restricted or virtually impossible. In other words it may be a one way switch.

13.5 Database Links

These will need changing as the standby databases become active.

14 Problems encountered.

14.1 ORA-16009 Error and RFS Warnings

RFS Warnings and ORA-16009 Errors in Alert Log

The following errors were encountered on an Oracle Enterprise 9.2.0.8 Data Guard system.

- symptom: Errors appears in alert.log on primary database
- symptom: RFS: Destination database mount ID mismatch
- symptom: RFS: client instance is standby database instead of primary
- symptom: RFS: Not using real application clusters
- symptom: RFS: Possible network disconnect with primary database
- symptom: Errors appear in alert log on standby database
- symptom: ARC0: Error 16009 Creating archive log file to 'ORCL'
- symptom: ORA-16009: remote archive log destination must be a STANDBY database
- symptom: LOG_ARCHIVE_DEST_2='SERVICE=ORCL LGWR SYNC AFFIRM' on the standby database
- symptom: LOG_ARCHIVE_DEST_2='SERVICE=STBY LGWR SYNC AFFIRM' on the primary database
- symptom: Standby redo log files are defined on the standby database
- cause: The standby redo log files are synchronously filled with redo from the primary database. When a log switch occurs on the primary database, those files are archived on the standby database before being applied on it. The archiving process on the standby database should only archive to the local disks on the standby database and not transmit them to the primary database.

The solution is to disable the remote archiving on the standby database.

alter system set log_archive_dest_2 = ''

14.2 Synchronisation due to log transfer gap

The problem is that the standby database is not able to operate in managed recovery mode whilst there is a gap in the archives produced by the primary database. It may be necessary to synchronize the primary and standby database.

1. Detection of the log gap primary and standby database.

   a) Use a sql statement.

   b) Check the alert files.

   c) Check for fal server process.

   d) Checking archive destinations.

2. Synchronize the primary and standby database.

   a) Make the missing archives available for the standby database.

   b) Recover the standby database automatic.

3. Check the log gap primary and standby database is solved.

a) Use a sql statement.

b) Check the alert files.

After correcting the log gap problem place the standby database in sustained managed recovery mode

1. Detection of the log gap primary and standby database.

a) Using a sql statement.

To find out which logs have not been received by this standby destination, issue the following query at the primary database:

```
SQL> SELECT local.thread#, local.sequence# from
   (select thread#, sequence# from v$archived_log where dest_id=1) local
   where
   local.sequence# not in
   (select sequence# from v$archived_log where dest_id=2 and
   thread# = local.thread#);

THREAD#  SEQUENCE#
-------- ----------
     1      521
     1      522
     1      523
     1      524
     1      525
```

Another possible statement to check this is (at the standby database):

```
SELECT thread#, low_sequence#, high_sequence#
from V$archive_gap;
```

Note: the sql statements give additional information (confirmation) of information of alert files. If these sql statements return rows it doesn't necessarily mean there's an actual gap! Always check the alert files as in 1b.

b) Check the alert files.

Alert file primary database:

```
Thu Jun 21 13:23:13 2001
ARC0: Beginning to archive log 1 thread 1 sequence 525
ARC0: Warning; LGWR is actively archiving destination LOG_ARCHIVE_DEST_2
ARC0: Transmitting activation ID 680595809 (28911161)
ARC0: Error 12541 connecting to standby host 'ssym_nlsu22.world'
ARC0: Error 12541 Creating archive log file to 'ssym_nlsu22.world'
ARC0: Completed archiving  log 1 thread 1 sequence 525
Thu Jun 21 13:40:21 2001
Thread 1 advanced to log sequence 527
 Current log# 1 seq# 527 mem# 0: /ots0/app/oracle/product/9.0.1/dbs/sroo/LOG1Oa
 Current log# 1 seq# 527 mem# 1: /ots0/app/oracle/product/9.0.1/dbs/sroo/LOG2Oa
Thu Jun 21 13:40:21 2001
ARC0: Beginning to archive log 2 thread 1 sequence 526
ARC0: Warning; LGWR is actively archiving destination LOG_ARCHIVE_DEST_2
ARC0: Transmitting activation ID 680595809 (28911161)
ARC0: Completed archiving  log 2 thread 1 sequence 526
```

Alert file standby:

```
$ tail -50 alert_ssy*|more
Thu Jun 21 13:34:53 2001
Media Recovery Start: Managed Standby Recovery
Successfully started datafile 1 media recovery
Datafile #1: '/ots0/app/oracle/product/9.0.1/dbs/sroo/backup/system09.dbf'
Media Recovery Log
Media Recovery Waiting for thread 1 seq# 522
Thu Jun 21 13:40:23 2001
```

```
Fetching gap sequence for thread 1, gap sequence 522-525
Trying FAL server:
Error fetching gap sequence, no FAL server specified
Thu Jun 21 13:40:38 2001
Failed to request gap sequence. Thread #: 1, gap sequence: 522-525
All FAL server has been attempted.
```

c) Fal server process.

A Fal server process is an Oracle server process running on the primary database servicing fal request from a fal client. An example of a request of a client are queueing requests to send archived redologfiles from a primary database to one or more standby databases.

d) Checking archive destinations.

Checking the archive destination of the primary database:

```
/home/server/sroo/test
$ ls
arch_1_521.arc   arch_1_523.arc   arch_1_525.arc
arch_1_522.arc   arch_1_524.arc   arch_1_526.arc   testsroo/
```

Checking the archive destination of the standby database:

/home/server/sroo/test/testsroo

```
$ ls
arch_1_526.arc
```

2 Synchronize the primary and standby database.

a) Make the missing archives available for the standby database.

Copy the missing archives from the gap to a location that can be reached by the standby database that is the standby_archive_dest.

```
$ pwd
  /home/server/sroo/test
$ cp *521.arc testsroo
$ cp *522.arc testsroo
$ cp *523.arc testsroo
$ cp *524.arc testsroo
$ cp *525.arc testsroo
```

b) Recover the standby database automatic.

Cancel the managed recovery

after cancel the standby recovery session shows:

```
SQL> alter database recover managed standby database cancel;
ORA-01547: warning: RECOVER succeeded but OPEN RESETLOGS would get error below
ORA-01152: file 1 was not restored from a sufficiently old backup
ORA-01110: data file 1:
'/ots0/app/oracle/product/9.0.0/dbs/sroo/backup/system09.dbf'
ORA-16037: user requested cancel of managed recovery operation
```

Then recover the standby database:

```
SQL> alter database recover automatic standby database;
ORA-00279: change 61729 generated at 06/21/2001 13:40:21 needed for thread 1
ORA-00289: suggestion : /home/server/sroo/test/testsroo/arch_1_527.arc
ORA-00280: change 61729 for thread 1 is in sequence #527
ORA-00278: log file '/home/server/sroo/test/testsroo/arch_1_527.arc' no longer
needed for this recovery
ORA-00308: cannot open archived log
'/home/server/sroo/test/testsroo/arch_1_527.arc'
ORA-27037: unable to obtain file status
SVR4 Error: 2: No such file or directory
Additional information: 3
Specify log: {<RET>=suggested | filename | AUTO | CANCEL}
 => cancel
```

```
Media recovery cancelled.
```

Set the standby database back to work:

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

3 Check the log gap primary and standby database is solved.

a) Using an sql statement.

To find out which logs have not been received by this standby destination, issue the following query at the primary database:

```
SQL> SELECT local.thread#, local.sequence# from
  (select thread#, sequence# from v$archived_log where dest_id=1) local
   where
  local.sequence# not in
  (select sequence# from v$archived_log where dest_id=2 and
  thread# = local.thread#);

=> no rows selected => OK!
```

Note: Even if this sql statement returns rows synchronisation may be successful - always check alert files (next item).

b) Checking the alert files.

Check alert files if archives are now picked up by standby database:

Checking alert file primary:

```
$ tail -50 alert_sy*
Thu Jun 21 13:23:13 2001
ARC0: Beginning to archive log 1 thread 1 sequence 525
ARC0: Warning; LGWR is actively archiving destination LOG_ARCHIVE_DEST_2
ARC0: Transmitting activation ID 680595809 (28911161)
ARC0: Error 12541 connecting to standby host 'ssym_nlsu22.world'
ARC0: Error 12541 Creating archive log file to 'ssym_nlsu22.world'
ARC0: Completed archiving  log 1 thread 1 sequence 525
Thu Jun 21 13:40:21 2001
Thread 1 advanced to log sequence 527
 Current log# 1 seq# 527 mem# 0: /ots0/app/oracle/product/9.0.1/dbs/sroo/LOG1Oa
 Current log# 1 seq# 527 mem# 1: /ots0/app/oracle/product/9.0.1/dbs/sroo/LOG2Oa
Thu Jun 21 13:40:21 2001
ARC0: Beginning to archive log 2 thread 1 sequence 526
ARC0: Warning; LGWR is actively archiving destination LOG_ARCHIVE_DEST_2
ARC0: Transmitting activation ID 680595809 (28911161)
ARC0: Completed archiving  log 2 thread 1 sequence 526
Thu Jun 21 13:56:37 2001
LGWR: Transmitting activation ID 680595809 (28911161)
LGWR: Beginning to archive log 2 thread 1 sequence 528
Thread 1 advanced to log sequence 528
 Current log# 2 seq# 528 mem# 0: /ots0/app/oracle/product/9.0.1/dbs/sroo/LOG3Oa
 Current log# 2 seq# 528 mem# 1: /ots0/app/oracle/product/9.0.1/dbs/sroo/LOG4Oa
Thu Jun 21 13:56:38 2001
ARC0: Beginning to archive log 1 thread 1 sequence 527
ARC0: Warning; LGWR is actively archiving destination LOG_ARCHIVE_DEST_2
ARC0: Completed archiving  log 1 thread 1 sequence 527
```

Checking alert file standby:

```
$ tail -50 alert_ssy*
Thu Jun 21 13:54:45 2001
ALTER DATABASE RECOVER automatic standby database;
Thu Jun 21 13:54:45 2001
Media Recovery Start
Successfully started datafile 1 media recovery
Datafile #1: '/ots0/app/oracle/product/9.0.1/dbs/sroo/backup/system09.dbf'
Media Recovery Log
Media Recovery Log /home/server/sroo/test/testsroo/arch_1_522.arc
Media Recovery Log /home/server/sroo/test/testsroo/arch_1_523.arc
```

```
Media Recovery Log /home/server/sroo/test/testsroo/arch_1_524.arc
Media Recovery Log /home/server/sroo/test/testsroo/arch_1_525.arc
Media Recovery Log /home/server/sroo/test/testsroo/arch_1_526.arc
Media Recovery Log /home/server/sroo/test/testsroo/arch_1_527.arc
ORA-279 signalled during: ALTER DATABASE RECOVER automatic standby database;
Thu Jun 21 13:55:01 2001
ALTER DATABASE RECOVER    CANCEL;
Media Recovery Cancelled
Completed: ALTER DATABASE RECOVER    CANCEL
Thu Jun 21 13:56:00 2001
ALTER DATABASE RECOVER managed standby database
Thu Jun 21 13:56:00 2001
Media Recovery Start: Managed Standby Recovery
Successfully started datafile 1 media recovery
Datafile #1: '/ots0/app/oracle/product/9.0.1/dbs/sroo/backup/system09.dbf'
Media Recovery Log
Media Recovery Waiting for thread 1 seq# 527
Thu Jun 21 13:56:45 2001
Media Recovery Log /home/server/sroo/test/testsroo/arch_1_527.arc
Media Recovery Waiting for thread 1 seq# 528
=> OK!
```

14.3 ORA-16166 LGWR timed out on Network Server 0

While archiving to a remote standby using LGWR ASYNC, you receive the following type of messages in the alert.log:

```
Tue Nov  6 09:43:17 2007
Timing out on NetServer 0 prod=7145,cons=6853,threshold=256
ORA-16166: LGWR timed out on Network Server 0 due to buffer full condition.
         No action is required since the log file transfer will be attempted via
 ARCH
Tue Nov  6 09:43:17 2007
Errors in file /home/oracle/admin/jdapmm/bdump/jdapmm_lgwr_224020.trc:
ORA-16166: LGWR network server failed to send remote message
LGWR: I/O error 16166 archiving log 1 to 'standby_jdapmm.world'
Tue Nov  6 09:43:17 2007
Errors in file /home/oracle/admin/jdapmm/bdump/jdapmm_lgwr_224020.trc:
ORA-16166: LGWR network server failed to send remote message1.
```

Increase the ASYNC buffer size to the maximum. As a best practice define your remote archive destination as such:

service=<tns alias> LGWR ASYNC=20480 reopen=15 max_failure=10 net_timeout=30

2. Validate that your network bandwidth is sufficient to support the redo rate. First use statspack on the primary database to determine your highest redo rate (Redo size from Statspack). When setting the Statspack snapshot interval, make sure the interval covers the peak load time and that the interval is not too long that it might reduce the average redo rate for the interval. Generally, 5-15 minute snapshots work well but depending on the time under study, it may be shorter. E.g. you could take a snapshot for a 1-hour interval and find the redo rate is 500 K/sec or 30 MB/hour.

However, when taking 5 minute snapshots over the hour you find that the redo rate ranged from 400 K/sec-1.2 MB/sec. The peak load snapshot with a rate of 1.2 MB/sec is what should be used for further diagnosis. Once you know your highest redo rate then use the following calculation to determine the amount of bandwidth necessary to sustain that redo rate:

```
Required bandwidth

        = ((Redo rate in bytes ps / .7) * 8) / 1,000,000

        = bandwidth in Mbps
```

3. Where possible, reduce the frequency of commits on the primary database.

When using LGWR to remotely archive in ASYNC mode, the LGWR process does not wait for each network I/O to complete before proceeding. This behaviour is made possible by the use of an intermediate process, known as a LGWR network server process (LNS) that performs the actual network I/O and waits for each network I/O to complete. Each LNS has a user configurable buffer that is used to accept outbound redo data from the LGWR. This is configured by specifying the size in 512 byte blocks on the ASYNC attribute in the archivelog destination parameter. For example ASYNC=2048 indicates a 1Mb buffer. As long as the LNS process is able to empty this buffer faster than the LGWR can fill it, the LGWR will never stall. If the LNS cannot keep up, then the buffer will become full and the LGWR will stall until either sufficient buffer space is freed up by a successful network transmission or a timeout occurs. When this buffer full condition is reached a message is printed to the alert.log.

By increasing the size of the buffer to the maximum of 10 meg (20480 512k blocks), and ensuring that the LNS process has enough bandwidth to empty the buffer in a timely fashion we can prevent the buffer from becoming full.

Notes:

When the `LGWR` and `ASYNC` attributes are specified, the log writer process writes to the local online redo log file, while the network server (LNSn) processes (one for each destination) asynchronously transmit the redo to remote destinations. The LGWR process continues processing the next request without waiting for the LNS network I/O to complete.

If redo transport services transmit redo data to multiple remote destinations, the LNSn processes (one for each destination) initiate the network I/O to all of the destinations in parallel.

When an online redo log file fills up, a log switch occurs and an archive process archives the log file locally, as usual.

- The `SYNC` attribute tells Data Guard to perform all network I/O synchronously for this standby, in conjunction with each write operation to the local online redo log file.
- The A`SYNC` attribute tells Data Guard to perform all network I/O asynchronously for this standby.
- The `AFFIRM` attribute insures that transactions are not acknowledged as committed by the primary database until the redo data necessary to recover the transactions has been written to the standby redo log at the remote destination.
- The `REOPEN=10` attribute specifies that the LGWR should not try to reconnect to a standby that was previously disconnected (due to a network failure, standby node failure, etc.) at the next log switch if that log switch occurs in less than 10 seconds. If not specified, the default wait for reopen is 300 seconds.
- The `NET_TIMEOUT=30` attribute specifies the number of seconds (30 seconds in this case), that the `LGWR` process on the primary system will wait for a reply from the standby server before terminating the network connection and continuing processing on the primary server. Following any loss of connection to the standby server, Data Guard continually monitors the status of the standby server and automatically resynchronizes the standby database with the primary database once it becomes available. The default value for `NET_TIMEOUT` is 180 seconds; the minimum value that can be set is 1 second.

A. Appendices

A.1 Sample Scripts

The following is the basis for a sample script to generate the appropriate commands in a file to be used for performing a hot backup of a database to create a standby instance. The generated script will need some editing before being run to ensure files are going to the correct destination.

```
PROCEDURE gen_script (inst_name IN VARCHAR2)
IS
   fname varchar2(32);
   rcnt integer := 0;

   cursor c1
   is
   select tablespace_name
   from dba_tablespaces
   where tablespace_name != 'TEMP';

   cursor c2 (tname in varchar2)
   is
   select file_name
   from dba_data_files
   where tablespace_name = tname;

BEGIN
   dbms_output.put_line('alter system archive log current;');
   dbms_output.put_line('alter system archive log current;');
   dbms_output.put_line('alter system archive log current;');
   FOR R1 IN c1 LOOP
      EXIT WHEN c1%NOTFOUND;
      rcnt := rcnt + 1;
      dbms_output.put_line('alter tablespace '||R1.tablespace_name||' begin backup;');
      FOR R2 in c2(R1.tablespace_name) LOOP
         fname := substr(r2.file_name,instr(r2.file_name,'/',-1)+1);
         dbms_output.put_line(
         '!rcp '||r2.file_name||' standby:/data/filesys/oradata/'||inst_name||'/'||fname);
      END LOOP;
      dbms_output.put_line('alter tablespace '||R1.tablespace_name||' end backup;');
   END LOOP;
   dbms_output.put_line('alter system archive log current;');
   dbms_output.put_line('alter system archive log current;');
   dbms_output.put_line('REM Do not forget to copy the archived log files over.!');
   dbms_output.put_line(
            'alter database create standby controlfile as ''/home/oracle/standby.ctl'';');
END gen_script;
```

A.2 Large databases

There is a small editing issue if there are a large number of data files to be relocated. In this situation a variation of the script above was created to resolve this.

A small database table was created to hold the translation required from the primary to the standby location. The table was created as follows:

```
CREATE TABLE SYSTEM.CONVERT
(
  PRIMARY_LOC  VARCHAR2(128 BYTE),
  STANDBY_LOC  VARCHAR2(128 BYTE)
)

TABLESPACE USERS;
```

This table was then populated with appropriate values for the primary_loc and the standby_loc.

```
INSERT INTO SYSTEM.CONVERT (PRIMARY_LOC, STANDBY_LOC)
VALUES (‘/filesysa/oradata/dbinst’,’/data/filesysb/oradata/dbinst’);
```

The main procedure was then modified to reference this table.

```
DECLARE
   PROCEDURE gen_script (inst_name IN VARCHAR2, stat in VARCHAR2 DEFAULT 'ONLINE')
   IS
      fname varchar2(32);
```

```
        rcnt integer := 0;
        dirin varchar2(128);
        dirout varchar2(128);

        cursor c1
        is
        select tablespace_name
        from dba_tablespaces
        where tablespace_name != 'TEMP'
        and status = stat;

        cursor c1a
        is
        select tablespace_name
        from dba_tablespaces
        where tablespace_name != 'TEMP'
        and status != stat;

        cursor c2 (tname in varchar2)
        is
        select file_name
        from dba_data_files
        where tablespace_name = tname;

        BEGIN

           select count(*)
           into rcnt
           from dba_tablespaces
           where tablespace_name != 'TEMP'
           and status = stat;

           IF rcnt != 0 THEN
              dbms_output.put_line('alter system archive log current;');
              dbms_output.put_line('alter system archive log current;');
              dbms_output.put_line('alter system archive log current;');
              FOR R1 IN c1 LOOP
                 EXIT WHEN c1%NOTFOUND;
                 rcnt := rcnt + 1;
                 dbms_output.put_line('alter tablespace '||R1.tablespace_name||' begin backup;');
                 FOR R2 in c2(R1.tablespace_name) LOOP
                    fname := substr(r2.file_name,instr(r2.file_name,'/',-1)+1);

                    dirin := substr(r2.file_name,1,instr(r2.file_name,'/',-1)-1);

                    begin
                       select standby_loc
                       into dirout
                       from system.convert
                       where primary_loc = dirin;
                    exception
                       when no_data_found then
                          dbms_output.put_line('Location not found: '||dirin);
                    end;

                    dbms_output.put_line('!rcp '||r2.file_name||'standby:'||dirout||'/'||fname);
                 END LOOP;
                 dbms_output.put_line('alter tablespace '||R1.tablespace_name||' end backup;');
              END LOOP;
              dbms_output.put_line('alter system archive log current;');
              dbms_output.put_line('alter system archive log current;');
              dbms_output.put_line('REM Do not forget to copy the archived log files over.!');
              dbms_output.put_line('alter database create standby controlfile as
''/home/oracle/standby.ctl'';');
           ELSE
              dbms_output.put_line('No tablespaces to backup.');
           END IF;
        END gen_script;
     BEGIN
        gen_script ('dbinst');
        --gen_script ('dbinst','READ ONLY');
     END;
```

A.3 Script to remove applied logs on standby system.

The following script has been used to clean out logs received from the primary database and applied successfully on the standby system. This ensures that the disk space on the standby system does not get used up disproportionately.

```
#
# Script to remove archive logs that have been applied to the standby database.
#
# G S Chapman 16th October 2007
#
```

```
. /home/oracle/.profile

export ORAENV_ASK=NO
for i in ikbprod tvclive mlive jdapmm
do
   ORACLE_SID=$i
   . oraenv

sqlplus -s "/ as sysdba" <<EOF
set echo off
set pagesize 0
spool arch_remove.lst
select 'rm -f '|| al.name
from v\$archived_log al, v\$log_history lh
where al.sequence# = lh.sequence#
and al.applied='YES'
and lh.sequence# is NOT NULL
and al.completion_time between sysdate-1 and sysdate-1/24;
spool off;
exit;
EOF

   # Now we have the file we can get rid of the last three lines
   # which are rubbish.
   # We will also keep the last five real lines as well.
   lines=`cat arch_remove.lst | wc -l`
   # echo $lines
   lines=`expr $lines + 0 `
   # echo $lines
   nlines=`expr $lines - 4 `
   # echo $lines $nlines
   # Check that we have more than 4 lines in the output log.
   if [ $lines -gt 4 ] ; then
      head -n $nlines arch_remove.lst > rm_logs.sh
      if [ -f rm_logs.sh ] ; then
         chmod 700 rm_logs.sh
         ./rm_logs.sh
      fi
   fi
   rm -f arch_remove.lst rm_logs.sh
done
```

A.4 Monitoring scripts

A.4.1 Data Guard Physical Standby Diagnostic Information

The following is from Metalink Note: 241438.1.

```
-- NAME: DG_phy_stby_diag.sql
-- ------------------------------------------------------------------------
-- AUTHOR:
--    Michael Smith - Oracle Support Services - DataServer Group
--    Copyright 2002, Oracle Corporation
-- ------------------------------------------------------------------------
-- PURPOSE:
--    This script is to be used to assist in collection information to help
--    troubeshoot Data Guard issues.
-- ------------------------------------------------------------------------
-- DISCLAIMER:
--    This script is provided for educational purposes only. It is NOT
--    supported by Oracle World Wide Technical Support.
--    The script has been tested and appears to work as intended.
--    You should always run new scripts on a test instance initially.
-- ------------------------------------------------------------------------
-- Script output is as follows:

set echo off
set feedback off
column timecol new_value timestamp
column spool_extension new_value suffix
select to_char(sysdate,'Mondd_hhmi') timecol,
'.out' spool_extension from sys.dual;
column output new_value dbname
select value || '_' output
from v$parameter where name = 'db_name';
spool dgdiag_phystby_&&dbname&&timestamp&&suffix
```

```
set lines 200
set pagesize 35
set trim on
set trims on
alter session set nls_date_format = 'MON-DD-YYYY HH24:MI:SS';
set feedback on
select to_char(sysdate) time from dual;

set echo on

--
-- ARCHIVER can be  (STOPPED | STARTED | FAILED) FAILED means that the archiver failed
-- to archive a -- log last time, but will try again within 5 minutes. LOG_SWITCH_WAIT
-- The ARCHIVE LOG/CLEAR LOG/CHECKPOINT event log switching is waiting for. Note that
-- if ALTER SYSTEM SWITCH LOGFILE is hung, but there is room in the current online
-- redo log, then value is NULL

column host_name format a20 tru
column version format a9 tru
select instance_name,host_name,version,archiver,log_switch_wait from v$instance;

-- The following select will give us the generic information about how this standby is
-- setup.  The database_role should be standby as that is what this script is intended
-- to be ran on.  If protection_level is different than protection_mode then for some
-- reason the mode listed in protection_mode experienced a need to downgrade.  Once the
-- error condition has been corrected the protection_level should match the protection_mode
-- after the next log switch.

column ROLE format a7 tru
select name,database_role,log_mode,controlfile_type,protection_mode,protection_level
from v$database;

-- Force logging is not mandatory but is recommended.  Supplemental logging should be enabled
-- on the standby if a logical standby is in the configuration. During normal
-- operations it is acceptable for SWITCHOVER_STATUS to be SESSIONS ACTIVE or NOT ALLOWED.

column force_logging format a13 tru
column remote_archive format a14 tru
column dataguard_broker format a16 tru
select force_logging,remote_archive,supplemental_log_data_pk,supplemental_log_data_ui,
switchover_status,dataguard_broker from v$database;

-- This query produces a list of all archive destinations and shows if they are enabled,
-- what process is servicing that destination, if the destination is local or remote,
-- and if remote what the current mount ID is. For a physical standby we should have at
-- least one remote destination that points the primary set but it should be deferred.

COLUMN destination FORMAT A35 WRAP
column process format a7
column archiver format a8
column ID format 99

select dest_id "ID",destination,status,target,
archiver,schedule,process,mountid
from v$archive_dest;

-- If the protection mode of the standby is set to anything higher than max performance
-- then we need to make sure the remote destination that points to the primary is set
-- with the correct options else we will have issues during switchover.

select dest_id,process,transmit_mode,async_blocks,
net_timeout,delay_mins,reopen_secs,register,binding
from v$archive_dest;

-- The following select will show any errors that occured the last time an attempt to
-- archive to the destination was attempted.  If ERROR is blank and status is VALID then
-- the archive completed correctly.

column error format a55 tru
select dest_id,status,error from v$archive_dest;

-- Determine if any error conditions have been reached by querying thev$dataguard_status
-- view (view only available in 9.2.0 and above):

column message format a80
select message, timestamp
from v$dataguard_status
where severity in ('Error','Fatal')
order by timestamp;

-- The following query is ran to get the status of the SRL's on the standby.  If the
-- primary is archiving with the LGWR process and SRL's are present (in the correct
-- number and size) then we should see a group# active.

select group#,sequence#,bytes,used,archived,status from v$standby_log;

-- The above SRL's should match in number and in size with the ORL's returned below:
```

```
select group#,thread#,sequence#,bytes,archived,status from v$log;

-- Query v$managed_standby to see the status of processes involved in the
-- configuration.

select process,status,client_process,sequence#,block#,active_agents,known_agents
from v$managed_standby;

-- Verify that the last sequence# received and the last sequence# applied to standby
-- database.

select max(al.sequence#) "Last Seq Recieved", max(lh.sequence#) "Last Seq Applied"
from v$archived_log al, v$log_history lh;

-- The V$ARCHIVE_GAP fixed view on a physical standby database only returns the next
-- gap that is currently blocking redo apply from continuing. After resolving the
-- identified gap and starting redo apply, query the V$ARCHIVE_GAP fixed view again
-- on the physical standby database to determine the next gap sequence, if there is
-- one.

select * from v$archive_gap;

-- Non-default init parameters.

set numwidth 5
column name format a30 tru
column value format a50 wra
select name, value
from v$parameter
where isdefault = 'FALSE';

spool off
```

A.4.2 Data Guard Primary Site Diagnostic Information

The following is from Metalink Note: 241374.1

```
-- NAME: dg_prim_diag.sql  (Run on PRIMARY with a LOGICAL or PHYSICAL STANDBY)
-- ---------------------------------------------------------------------------
--    Copyright 2002, Oracle Corporation
-- LAST UPDATED: 2/23/04
--
-- Usage: @dg_prim_diag
-- ---------------------------------------------------------------------------
-- PURPOSE:
--    This script is to be used to assist in collection information to help
--    troubeshoot Data Guard issues with an emphasis on Logical Standby.
-- ---------------------------------------------------------------------------
-- DISCLAIMER:
--    This script is provided for educational purposes only. It is NOT
--    supported by Oracle World Wide Technical Support.
--    The script has been tested and appears to work as intended.
--    You should always run new scripts on a test instance initially.
-- ---------------------------------------------------------------------------
-- Script output is as follows:

set echo off
set feedback off
column timecol new_value timestamp
column spool_extension new_value suffix
select to_char(sysdate,'Mondd_hhmi') timecol,
'.out' spool_extension from sys.dual;
column output new_value dbname
select value || '_' output
from v$parameter where name = 'db_name';
spool dg_prim_diag_&&dbname&&timestamp&&suffix
set linesize 79
set pagesize 35
set trim on
set trims on
alter session set nls_date_format = 'MON-DD-YYYY HH24:MI:SS';
set feedback on
select to_char(sysdate) time from dual;

set echo on

-- In the following the database_role should be primary as that is what
-- this script is intended to be run on.  If protection_level is different
-- than protection_mode then for some reason the mode listed in
-- protection_mode experienced a need to downgrade.  Once the error
-- condition has been corrected the protection_level should match the
-- protection_mode after the next log switch.

column role format a7 tru
column name format a10 wrap

select name,database_role role,log_mode,
```

```
       protection_mode,protection_level
from v$database;

-- ARCHIVER can be (STOPPED | STARTED | FAILED). FAILED means that the
-- archiver failed to archive a log last time, but will try again within 5
-- minutes. LOG_SWITCH_WAIT The ARCHIVE LOG/CLEAR LOG/CHECKPOINT event log
-- switching is waiting for.  Note that if ALTER SYSTEM SWITCH LOGFILE is
-- hung, but there is room in the current online redo log, then value is
-- NULL

column host_name format a20 tru
column version format a9 tru

select instance_name,host_name,version,archiver,log_switch_wait
from v$instance;

-- The following query give us information about catpatch.
-- This way we can tell if the procedure doesn't match the image.

select version, modified, status from dba_registry
where comp_id = 'CATPROC';

-- Force logging is not mandatory but is recommended.  Supplemental
-- logging must be enabled if the standby associated with this primary is
-- a logical standby. During normal operations it is acceptable for
-- SWITCHOVER_STATUS to be SESSIONS ACTIVE or TO STANDBY.

column force_logging format a13 tru
column remote_archive format a14 tru
column dataguard_broker format a16 tru

select force_logging,remote_archive,
       supplemental_log_data_pk,supplemental_log_data_ui,
       switchover_status,dataguard_broker
from v$database;

-- This query produces a list of all archive destinations.  It shows if
-- they are enabled, what process is servicing that destination, if the
-- destination is local or remote, and if remote what the current mount ID
-- is.

column destination format a35 wrap
column process format a7
column archiver format a8
column ID format 99
column mid format 99

select dest_id "ID",destination,status,target,
       schedule,process,mountid  mid
from v$archive_dest order by dest_id;

-- This select will give further detail on the destinations as to what
-- options have been set.  Register indicates whether or not the archived
-- redo log is registered in the remote destination control file.

set numwidth 8
column ID format 99

select dest_id "ID",archiver,transmit_mode,affirm,async_blocks async,
       net_timeout net_time,delay_mins delay,reopen_secs reopen,
       register,binding
from v$archive_dest order by dest_id;

-- The following select will show any errors that occured the last time
-- an attempt to archive to the destination was attempted.  If ERROR is
-- blank and status is VALID then the archive completed correctly.

column error format a55 wrap

select dest_id,status,error from v$archive_dest;

-- The query below will determine if any error conditions have been
-- reached by querying the v$dataguard_status view (view only available in
-- 9.2.0 and above):

column message format a80

select message, timestamp
from v$dataguard_status
where severity in ('Error','Fatal')
order by timestamp;

-- The following query will determine the current sequence number
-- and the last sequence archived.  If you are remotely archiving
-- using the LGWR process then the archived sequence should be one
-- higher than the current sequence.  If remotely archiving using the
-- ARCH process then the archived sequence should be equal to the
-- current sequence.  The applied sequence information is updated at
```

```
-- log switch time.

select ads.dest_id,max(sequence#) "Current Sequence",
       max(log_sequence) "Last Archived"
from v$archived_log al, v$archive_dest ad, v$archive_dest_status ads
where ad.dest_id=al.dest_id
and al.dest_id=ads.dest_id
group by ads.dest_id;

-- The following select will attempt to gather as much information as
-- possible from the standby.  SRLs are not supported with Logical Standby
-- until Version 10.1.

set numwidth 8
column ID format 99
column "SRLs" format 99
column Active format 99

select dest_id id,database_mode db_mode,recovery_mode,
       protection_mode,standby_logfile_count "SRLs",
       standby_logfile_active ACTIVE,
       archived_seq#
from v$archive_dest_status;

-- Query v$managed_standby to see the status of processes involved in
-- the shipping redo on this system.  Does not include processes needed to
-- apply redo.

select process,status,client_process,sequence#
from v$managed_standby;

-- The following query is run on the primary to see if SRL's have been
-- created in preparation for switchover.

select group#,sequence#,bytes from v$standby_log;

-- The above SRL's should match in number and in size with the ORL's
-- returned below:

select group#,thread#,sequence#,bytes,archived,status from v$log;

-- Non-default init parameters.

set numwidth 5
column name format a30 tru
column value format a48 wra
select name, value
from v$parameter
where isdefault = 'FALSE';

spool off
```

A.4.3 Alternative Standby Monitoring script

```
#!/bin/ksh

trap "" TTIN TTOU
export ORACLE_HOME=/apps/oracle/product/9.2.0.5
export ORACLE_SID=DGPERF1
export LD_LIBRARY_PATH=/apps/oracle/product/9.2.0.5/lib
export TNS_ADMIN=/apps/oracle/admin/network/admin
export PATH=/apps/oracle/product/9.2.0.5/bin:/usr/bin

#while true
#do
sqlplus '/ as sysdba' << END
spool $ORACLE_SID/dbstats.`date +%b%d_%T`
set pagesize 10000 echo off feedback off TERMOUT OFF
set trimspool on linesize 3000

REM Verify the state of the DG processes for more detailed
REM analysis if required
select process, status , thread#, sequence#, blocks from v\$managed_standby;
select max(sequence#), thread# from v\$log_history group by thread#;
column event format a35
column p1text format a20
column p2text format a20

REM Obtain session wait information for more detailed
REM analysis if required
select sid, event, p1, p1text, p2, p2text
from v\$session_wait
where wait_time !=0 and
event not in ('rdbms ipc message','smon timer')
order by wait_time desc;
```

```
REM Obtain file READ I/O and WRITE I/O times to ensure
REM there's no IO bottlenecks on the standby. Should
REM be similar to production I/O times.
column datafile format A72
column tspace format A30
select fs.*, df.name datafile, ts.name tspace
from v\$filestat fs, v\$datafile df, v\$tablespace ts
where fs.file#=df.file#
and df.ts#=ts.ts#
and PHYWRTS >0
order by writetim desc;

REM Obtain top system wait events. Leveraged to get
REM average log file parallel write times on the standby.
select * from v\$system_event where time_waited > 100
order by time_waited desc;

REM Obtain sysstat detailed statistics for detailed
REM analysis if required.
select name, value from v\$sysstat where name like 'recovery%';
spool off
exit
END

# sleep 60 # interval to obtain statistics
# done
exit 0
```