Script to Collect Data Guard Physical and Active Standby Diagnostic Information for Version 10g and above (Including RAC) (Doc ID 1577406.1)

APPLIES TO:

Oracle Database - Ent Edition - Version 10.1.0.2 and later
Information in this document applies to any platform.

MAIN CONTENT

```
-- NAME: new_dg_psby_diag.sql
--
-- Copyright 2002, Oracle Corporation
--
-- LAST UPDATED: 02-Sep-2015
--
-- Usage: @new_dg_psby_diag
--
-- (Run from sqlplus on PHYSICAL STANDBY, ACTIVE S
--
-- PURPOSE:
--
--  This script is to be used to assist in the collection of information to help
--  troubleshoot Data Guard issues involving a Physical or Active 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_v  lue timestamp
column spool_extension new_v  lue suffix
SELECT TO_CHAR(sysdate,'yyyymmdd_hh24mi') timecol, '.html' spool_extension FROM dual;
column output new_v  lue dbname
SELECT v  lue || '_' output FROM v$parameter WHERE name = 'db_unique_name';
spool new_dg_psby_diag_&&dbname&&timestamp&&suffix
set linesize 2000
set pagesize 50000
set numformat 999999999999999
set trim on
set trims on
set markup html on
set markup html entmap off
set feedback on

ALTER SESSION SET nls_date_format = 'DD-MON-YYYY HH24:MI:SS';
SELECT TO_CHAR(sysdate) time FROM dual;

set echo on
```

```
-- 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 run on.
-- PLATFORM_ID should match the PLATFORM_ID of the primary or conform to the supported options in
-- Note: 413484.1 Data Guard Support for Heterogeneous Primary and Physical Standbys in Same Data Guard Configuration.
-- FLASHBACK can be YES (recommended) or NO.
-- If PROTECTION_LEVEL is different from 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.

SELECT database_role role, name, db_unique_name, platform_id, open_mode, log_mode, flashback_on, protection_mode, protection_level FROM v$database;

-- FORCE_LOGGING is not mandatory but is recommended.
-- REMOTE_ARCHIVE should be ENABLE.
-- SUPPLEMENTAL_LOG_DATA_PK and SUPPLEMENTAL_LOG_DATA_UI must be enabled if this standby is associated with a primary that has a logical standby.
-- During normal operations it is acceptable for SWITCHOVER_STATUS to be NOT ALLOWED.
-- DATAGUARD_BROKER can be ENABLED (recommended) or DIS      ED.

column force_logging format a13 tru
column supplemental_log_data_pk format a24 tru
column supplemental_log_data_ui format a24 tru

SELECT force_logging, remote_archive, supplemental_log_data_pk, supplemental_log_data_ui, switchover_status, dataguard_broker  FROM v$database;

-- Check how many threads are enabled and started for this database. If the number of instances below does not match then not all instances are up.

SELECT thread#, instance, status FROM v$thread;

-- The number of instances returned below is the number currently running.  If it does not match the number returned in Threads above then not all instances are up.
-- VERSION should match the version from the primary 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 the value is NULL.

column host_name format a32 wrap

SELECT thread#, instance_name, host_name, v         , archiver, log_switch_wait FROM gv$instance ORDER BY thread#;

-- Check the number and size of online redo logs on each thread.

SELECT thread#, group#, sequence#, bytes, archived ,status FROM v$log ORDER BY thread#, group#;

-- The following query is run to see if standby redo logs have been created.
-- The standby redo logs should be the same size as the online redo logs.
-- There should be (( # of online logs per thread + 1) * # of threads) standby redo logs.
-- A value of 0 for the thread# means the log has never been allocated.

SELECT thread#, group#, sequence#, bytes, archived, status FROM v$standby_log order by thread#, group#;

-- This query produces a list of defined 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.
```

```
-- For a physical standby we should have at least one remote destination that points the primary set.

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

SELECT thread#, dest_id, destination, gvad.status, target, schedule, process, mountid mid FROM gv$archive_dest gvad, gv$instance gvi WHERE gvad.inst_id = gvi.inst_id AND destination is NOT NULL ORDER BY thread#, dest_id;

-- 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.

set numwidth 8
column archiver format a8
column ID format 99
column error format a55 wrap

SELECT thread#, dest_id, gvad.archiver, transmit_mode, affirm, async_blocks, net_timeout, delay_mins, reopen_secs reopen, register, binding FROM gv$archive_dest gvad, gv$instance gvi WHERE gvad.inst_id = gvi.inst_id AND destination is NOT NULL ORDER BY thread#, 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.

SELECT thread#, dest_id, gvad.status, error FROM gv$archive_dest gvad, gv$instance gvi WHERE gvad.inst_id = gvi.inst_id AND destination is NOT NULL ORDER BY thread#, dest_id;

-- 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 timestamp, gvi.thread#, message FROM gv$dataguard_status gvds, gv$instance gvi WHERE gvds.inst_id = gvi.inst_id AND severity in ('Error','Fatal') ORDER BY timestamp, thread#;

-- Query gv$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 thread#, process, pid, status, client_process, client_pid, sequence#, block#, active_agents, known_agents FROM gv$managed_standby ORDER BY thread#, process;

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

SELECT al.thrd "Thread", almax "Last Seq Received", lhmax "Last Seq Applied" FROM (select thread# thrd, MAX(sequence#) almax FROM v$archived_log WHERE resetlogs_change#=(SELECT resetlogs_change# FROM v$database) GROUP BY thread#) al, (SELECT thread# thrd, MAX(sequence#) lhmax FROM v$log_history WHERE resetlogs_change#=(SELECT resetlogs_change# FROM v$database) GROUP BY thread#) lh WHERE al.thrd = lh.thrd;

-- Check the transport lag and apply lag from the V$DATAGUARD_S    TS view.  This is only relevant when LGWR log transport and real time apply are in use.

SELECT * FROM v$dataguard_stats WHERE name LIKE '%lag%';

-- Check how often and how far the apply lags.

SELECT name, time, unit, count, TO_DATE(last_time_updated, 'MM/DD/YYYY HH24:MI:SS') FROM v$standby_event_histogram ORDER BY unit DESC, time;
```

```
-- 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                E_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.
-- For a RAC DB Thread# = * means the value is the same for all threads (SID=*)
-- Threads with different values are shown with their individual thread# and values.

column num noprint

SELECT num, '*' "THREAD#", name, value FROM v$PARAMETER WHERE NUM IN (SELECT num FROM v$parameter
WHERE (isdefault = 'FALSE' OR ismodified <> 'FALSE') AND name NOT LIKE 'nls%'
MINUS
SELECT num FROM gv$parameter gvp, gv$instance gvi WHERE num IN (SELECT DISTINCT gvpa.num FROM
gv$parameter gvpa, gv$parameter gvpb WHERE gvpa.num = gvpb.num AND  gvpa.value <> gvpb.value AND
(gvpa.isdefault = 'FALSE' OR gvpa.ismodified <> 'FALSE') AND gvpa.name NOT LIKE 'nls%') AND gvi.inst_id =
gvp.inst_id  AND (gvp.isdefault = 'FALSE' OR gvp.ismodified <> 'FALSE') AND gvp.name NOT LIKE 'nls%')
UNION
SELECT num, TO_CHAR(thread#) "THREAD#", name, value FROM gv$parameter gvp, gv$instance gvi WHERE num
IN (SELECT DIS  INCT gvpa.num FROM gv$parameter gvpa, gv$parameter gvpb WHERE gvpa.num = gvpb.num
AND gvpa.v  lue <> gvpb.v  lue AND (gvpa.isdefault = 'FALSE' OR gvpa.ismodified <> 'FALSE') AND gvp.name NOT
LIKE 'nls%') AND gvi.inst_id = gvp.inst_id  AND (gvp.isdefault = 'FALSE' OR gvp.ismodified <> 'FALSE') AND
gvp.name NOT LIKE 'nls%' ORDER BY 1, 2;

spool off
set markup html off entmap on
set echo on
```