Oracle GRID Center

Batch Processing in Disaster Recovery Configurations

Best Practices for Oracle Data Guard

Creation Date: October 20, 2008
Version: 1.0

ORACLE

HITACHI
Inspire the Next

1. Introduction

IT systems and the data they contain have grown increasingly critical in the operations of enterprises today. Corporations must protect important business data, including sensitive customer information, while being able to quickly restore operations and minimize interruption of service in the event of system failures. This capability is required regardless of the nature or scope of a failure event, ranging from operator error and system or software failures that can cause a database to fail, to natural disasters that impact large geographic areas causing entire sites to fail.

Oracle Corporation Japan and Hitachi Ltd. are engaged in an ongoing series of verification testing of Business Continuity Management (BCM) platform solutions technologies at the Oracle GRID Center using Hitachi hardware and Oracle Database/Oracle Fusion Middleware. For disaster recovery—a key BCM element—verification tests have already confirmed operations and primary features for a disaster recovery environment, configured with Hitachi BladeSymphony high-reliability blade servers and Oracle Data Guard (disaster recovery feature provided in Oracle Database), under simulated disaster conditions. The findings of this initial analysis have been published in a technical white paper on the Oracle Technology Network.[1]

In the new series of verification tests described in this white paper, the Oracle GRID Center has looked more closely at typical operational activities in a production data center where Data Guard is in use. This paper focuses on considerations and performance optimizations for batch processing done to create or update large volumes of data in an Oracle Data Guard environment. The paper confirms the effectiveness of the new redo transport compression features in Oracle Database 11g for certain environments and documents operational best practices for its use.

---

[1] Hitachi Ltd. / Oracle Japan GRID Center Validation Report on Active Data Guard 11g with RMAN Network Duplicate, Snapshot Standby, Apply Performance and Fast-Start Failover
http://www.oracle.com/technology/deploy/availability/htdocs/maa.htm

2. Contents

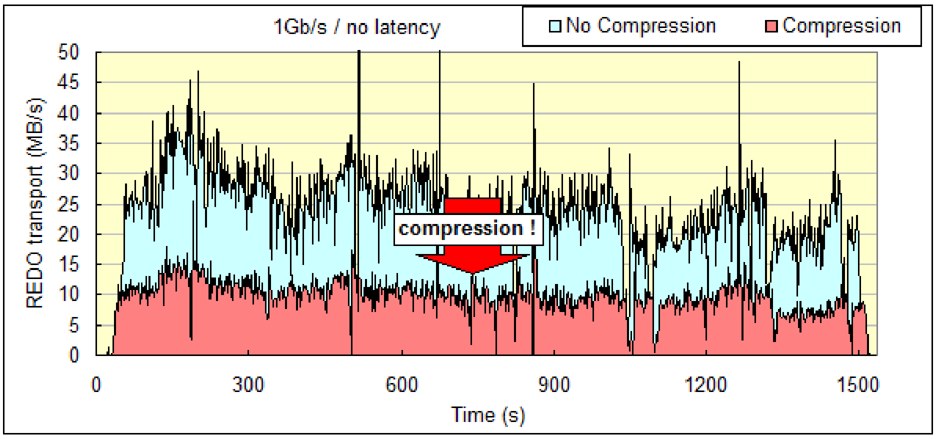
3. Executive Summary

This white paper describes important points for batch processing done to create large volumes of data in a disaster recovery environment configured with Oracle Real Application Clusters and Oracle Data Guard on Hitachi BladeSymphony servers. It also discusses the results of verification testing of effective tuning methods.

Oracle Data Guard synchronizes data between a primary database and one or more standby databases by transporting redo. Batch processing that updates large data volumes also generates correspondingly high redo volumes. Network bandwidth between primary and standby sites, while very adequate for normal processing, may be unable to handle the peak redo volumes generated during such batch processing. If this state persists for long periods where redo transport is not able to achieve the required throughput, **the resulting time lag of the newest data between primary and standby sites will most likely exceed your desired recovery point objective (RPO) – the amount of data vulnerable to loss if the primary site were to suddenly fail.**

The simplest way to solve this problem is to ensure adequate network bandwidth. However, securing adequate bandwidth (e.g., 1 Gb/s) for a redo transport network can be both expensive, and difficult for configurations in which standby databases are located at great distances from the primary database for maximum protection when a disaster strikes.

Redo transport compression, a feature of the Oracle Advanced Compression Option, is a new Oracle Database 11g capability that enables Oracle to compress redo as it is transported between primary and standby sites. Under the right circumstances it can be a very cost effective way to achieve RPO objectives by avoiding the need to procure additional network bandwidth in order to accommodate peaks in redo volume. Graph 3-1 compares volumes of transported data with and without redo compression under identical batch processing (data load) conditions. The graph shows that **compression can cut network volume by more than half.**

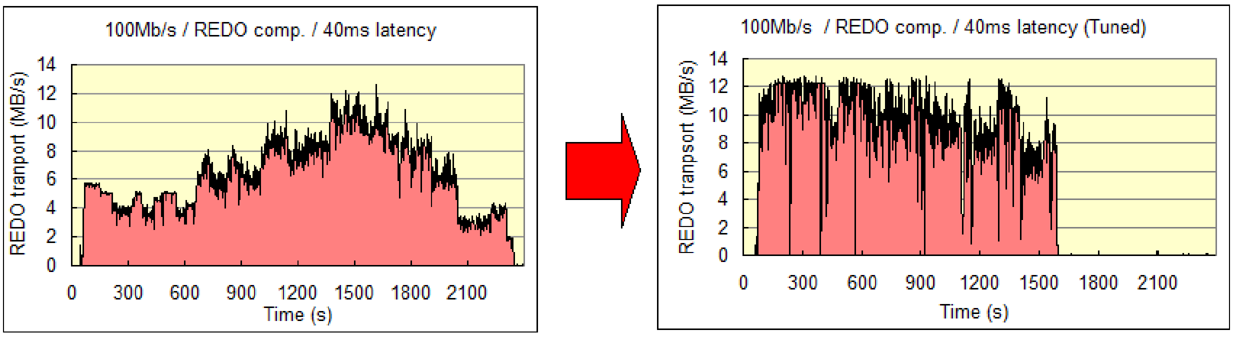


Graph 3-1: Comparison of volumes of transported redo with and without compression

Based on a verification scenario prepared for this verification testing, we confirmed that **redo**

compression reduced time lags between the primary database and standby database and **maintained data protection at virtually no cost in batch processing performance overhead**.

When standby databases are established at great distances from the primary database, system administrators must pay attention to transport delays and reduced redo transport efficiency. Reduced transport efficiency increases the time lag between primary and standby databases, in some cases barring effective data protection. Oracle Corporation provides Maximum Availability Architecture (MAA) best practices for resolving transport efficiency issues arising from transport delays by editing and tuning network-related parameters in Oracle Database[2]. Graph 3-2 shows changes in the volume of transported redo generated by batch processing (data load). The graph to the left shows measurements without tuning; the graph to the right shows measurements after tuning. **Tuning significantly improves transport efficiency**.



Graph 3-2: Network tuning efficacy

These results confirmed the potential value of utilizing the Advanced Compression Option for handling peaks in redo transport volume that accompany batch processing, and the benefits of following MAA best practices for network configuration when running Oracle Data Guard on Hitachi BladeSymphony servers.

[2] Data Guard Redo Transport and Network Best Practices http://www.oracle.com/technology/deploy/availability/htdocs/maa.htm

4. Batch Processing

Batch processing generally refers to computer processes for executing a series of programs (jobs) sequentially. Batch processing is used to perform tasks performed at regular intervals, such as payroll accounting or sales data aggregation—often during the night or on weekends, when computer resources are not being used for other purposes.

Shown below are some general requirements for batch processing:

1. Completion of all processes within the specified timeframe (batch window)
   Example: Order management system
   - Batch processing time: 5 hours (24:00 to 5:00)
   - Processing specifics: Sales calculation ⇒ Inventory calculation ⇒ Order sheet production
   - Routine business start time: 6:00

2. Establishment of troubleshooting method
   Example: Batch server clustering, retry setting by job tool

3. Scheduling of multiple jobs in specified sequence
   Example: Scheduling and monitoring by job tool

4. Efficient use of machine resources

Batch processing design is based on the requirements of tasks performed by the system in question. In batch processing, databases are subjected to long processing hours involving inserting, updating, aggregating, and querying large volumes of data. This means batch processing requires high performance and high availability from databases to ensure the specified tasks are completed within the specified batch window.

5. Purpose of Verification

The verification tests described in this white paper seek to confirm the following issues, focusing on batch processing in an Oracle Data Guard environment.

5.1 Items to be considered in batch processing in an Oracle Data Guard environment

Oracle Data Guard (DG) transports redo—the information needed to recover Oracle transactions—from a production database (primary database) to one or more standby databases. The standby database applies the redo data as it is received, enabling compete data synchronization with the primary database.

Batch processing generating large volumes of updates to be applied to the primary database also generates large volumes of redo, increasing the redo volume transported to a standby database. For Oracle Data Guard environments designed as disaster countermeasures, standby databases are often located at great distances from the primary databases, in many cases making it costly to secure adequate redo transport network bandwidth to handle peak workloads. In such cases, the volume of redo generated during batch processing can exceed the network transport efficiency over sustained periods, resulting in turn in problems for data protection and batch processing performance. In our verification tests, we performed batch processing in which the volume of generated redo exceeded network bandwidth to assess the effects on batch processing unique to Oracle Data Guard environments.

5.2 Effect of redo compression feature

The Advanced Compression Option of Oracle Database 11g enables redo transport compression functionality. When the redo compression feature is enabled, redo data is compressed before transport by Oracle Data Guard. This reduces the required bandwidth on the network used for redo transport and can shortens the time required to transport a given volume of redo. On the other hand, redo transport compression will also consume additional CPU resources. In our verification tests, we used batch processing that generated large volumes of redo and investigated impact of utilizing redo transport compression on both network throughput and primary database performance.

5.3 Network tuning results

Both network bandwidth and round-trip network latency (RTT) must be considered in redo transport to standby databases located at a distance. RTT latency can reduce transport efficiency, preventing effective use of network bandwidth. Testing also examined the impact of tuning in a network environment subject to high RTT latency.

6. Verification System Configuration

Figure 6-1 shows the configuration of the system used in our verification testing. The load client machines were connected to the database server via a public network. The bandwidth of the public network was 1 Gb/s.

We configured the Physical Standby Database. An independent network was configured to transport redo data from the primary database to the standby database. A network simulator was inserted during redo transport to control bandwidth and RTT network latency.

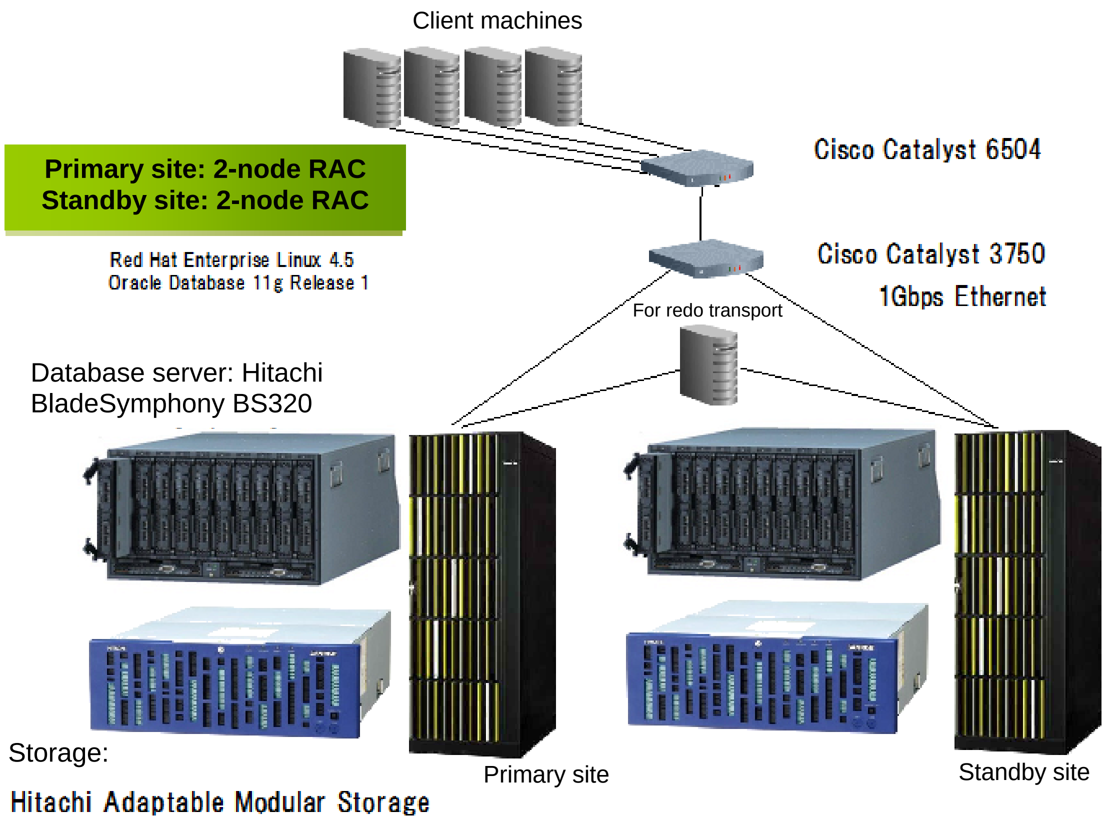


Figure 6-1: Verification system configuration

6.1.1 Hardware used

- Database server

Model	Hitachi BladeSymphony BS320 x 4 blades
CPU	Dual-Core Intel® Xeon® processor 3 GHz 2 sockets/blade
Memory	8 GB

- Client machine

Model	Intel White Box, 3 units
CPU	Dual-Core Intel® Xeon® processor 2.66 GHz 2 socket/server
Memory	4 GB

- Line simulator

Model	Hitachi BladeSymphony BS320 x 1 blade
CPU	Dual-Core Intel® Xeon® processor 3 GHz 2 sockets/blade
Memory	8 GB

- Storage

Model	Hitachi Adaptable Modular Storage (AMS)
Hard disk	144 GB x 28 HDD (+2 HDD as spare)
RAID group configuration	RAID5(2D+1P) x 8 (for Oracle Database; 4 RAID groups each for primary and secondary databases)

6.1.2 Software used

- Database server

OS	Red Hat Enterprise Linux 4.5
Oracle	Oracle Database 11g Release 1 (11.1.0.6) Enterprise Edition Oracle Real Application Clusters Oracle Active Data Guard Oracle Advanced Compression Oracle Partitioning

- Client machine

OS	Red Hat Enterprise Linux 4 Update 3
Oracle	Oracle Client 10g Release 2 (10.2)

- Line simulator

OS	Red Hat Enterprise Linux 4.5
Network control feature	netem tc-tbf

7. Verification Scenario and Confirmation Items

The following describes the scenario used in our verification testing, as well as validation results and required settings.

7.1 Processing sequence

We performed batch processing, OLTP processing, and read-only queries (simulating catalog browsing and reporting activities) using the Oracle Data Guard configuration established for verification testing. Figure 7-1 shows the schematics of the processes performed in the primary and standby databases.

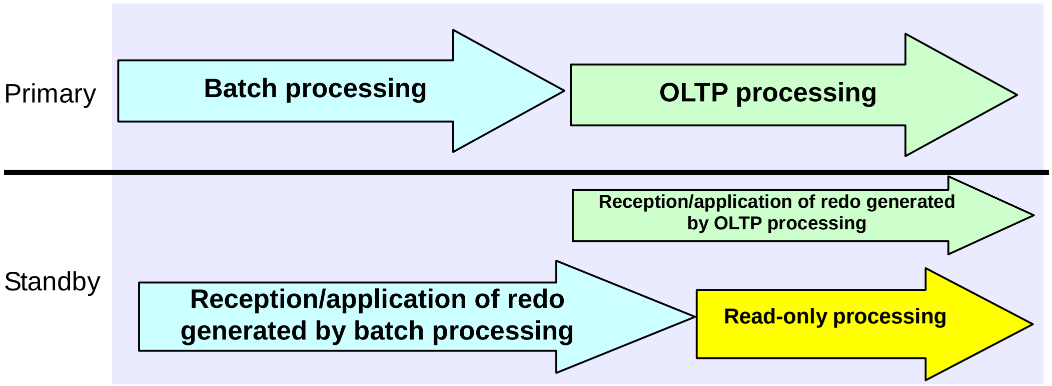


Figure 7-1: Processing sequence

Each process is described in detail below.

7.1.1 Batch processing

The batch processing included data loading, gathering statistics, and creating indexes to produce analysis data in Oracle Database. Table 7-1 provides a summary of processing details.

Process sequence	Process name	Process description
1	Data loading	This process loaded data totaling approximately 30 GB from multiple tables in flat files to Oracle Database. This process was implemented by parallel direct load insert using an external table feature.
2	Gathering statistics	This process collected statistical data on physical storage characteristics of tables containing loaded data to allow Oracle Database to select the most suitable data access

		path for analysis. This process was implemented using the DBMS_STATS package
3	Creating indexes	This process produced indexes in a selected table containing loaded data. Indexing was implemented in parallel while the automatic collection of statistical data remained effective.

Table 7-1: Details of batch processing

Batch processing was performed under the following two conditions:

(1) Enabling redo transport during batch processing
This case represents the ordinary method of operating Oracle Data Guard. In our verification tests, all scenarios in which redo transport was enabled were set to asynchronous transport. Depending on the volume of update data produced by batch processing, the volume of redo to be transported can exceed network bandwidth creating a time lag between the primary and standby databases causing data to be unprotected.

(2) Suspending redo transport during batch processing
In this case, redo transport is suspended during batch processing, allowing us to disregard the effects of the Oracle Data Guard configuration on batch processing performance. The redo generated by batch processing is automatically resolved as a gap when redo transport resumes after completion of batch processing, The time lag increases the entire time redo transport is suspended and then diminishes once the automatic gap resolution process begins.

7.1.2 OLTP processing

OLTP processing starts after batch processing completion. This operational sequence is based on an operational scenario in which routine business processing resumes after the completion of nighttime batch processing. In our tests, OLTP processing simulated workload generated by an online shopping site.

In the scenario where redo transport was suspended during batch processing, the redo generated by current OLTP processing is transmitted at the same time that the automatic gap resolution process is transmitting the redo generated during batch processing in order to eliminate the time lag as quickly as possible. (Note: the Data Guard will always apply transactions in SCN order).

7.1.3 Read-only Processing

The standby database was utilized for read-only access to data created by batch processing to simulate activities such as catalog browsing or reporting. Our scenario made the standby open for read-only access as soon as all the redo from batch processing had been applied to the standby database, and while the standby database continued to apply updates from OLTP transactions. The Oracle Active Data Guard Option, another new Oracle Database 11g capability, makes it possible for a physical standby database to be open read-only while updates received from the primary database continue to be applied. This made it possible to offload read-only workloads from the primary database to optimize OLTP performance while making efficient use of server resources at the disaster recovery site.

There are a couple of notable points in this scenario:

- There are actually two batch windows. The first window impacts OLTP processing on the primary database (deferred until batch processing completes on the primary database). The second window impacts read-only access on the standby database (deferred until the redo generated in the primary database by batch processing is applied to the standby database). This is illustrated in Figure 7-2.
- In an ideal operational scenario, batch processing completes as quickly as possible on the primary database to minimize the time that the primary is unavailable for OLTP processing. Likewise, redo transport is able to keep pace with batch processing so that there is no transport lag impacting either data protection, or how quickly the standby database is available for read-only access.

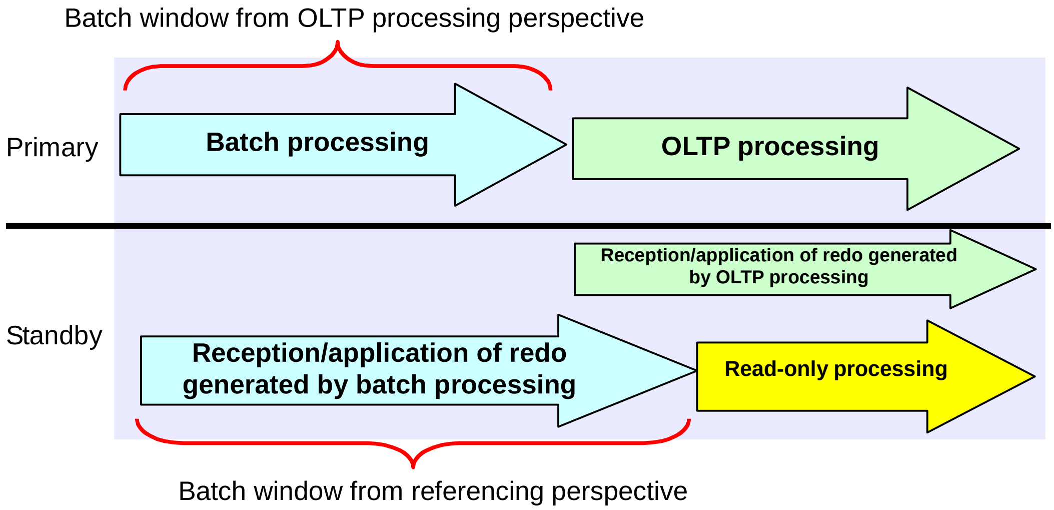


Figure 7-2: Batch windows

Our verification tests focused on processes executed during these two batch windows.

7.2 Redo compression setting

Redo transport compression is a feature of the Oracle Advanced Compression Option, a new enterprise edition database option introduced with Oracle Database 11g. This feature compresses redo transported from a primary database to a standby database. Redo transport compression offers the following configuration options:

7.2.1 Gap redo compression

This option compresses redo data transmitted during the Data Guard automatic gap resolution process. It compresses only the redo transmitted to resolve a gap (caused during a network or standby outage, or in one of our validation scenarios, caused by suspension of redo transport during batch processing). Enable redo transport compression for gaps by setting TRUE in the compression attribute of log_archive_dest_n; the initialization parameter that specifies the destination and transport method for redo transport in the primary database.

7.2.2 Asynchronous redo compression

This option compresses the redo to be transported by asynchronous (ASYNC) transport under normal conditions, in addition to redo compressed during gap resolution. ASYNC redo transport is the method of transport specified by Data Guard Maximum Performance mode.

> For information on setting the asynchronous redo compression, refer to MetaLink note: 729551.1.

Redo compression is initiated in the primary database during redo transport. The redo is uncompressed as it is received by the standby database. Note that implementing redo compression requires consideration of CPU overhead resulting from the two operations.

7.3 Network tuning

In communications with a server established at a distant location to ensure disaster recovery, communication delays prolong the network round trip time (RTT) and can reduce transport efficiency. A similar phenomenon can also arise during Data Guard redo transport. While Data Guard 11*g* enhancements to ASYNC redo transport have significantly reduced the impact of RTT network latency, a small amount of tuning will enhance redo transport efficiency for high latency and/or high bandwidth networks when shipping a high volume of redo. This is due to the influence that latency and bandwidth have on the calculation of bandwidth-delay product (BDP), and thus the optimum value for TCP send and receive buffer sizes as described in Table 7-2. Note that all network tuning during our verification testing only required editing parameters in Oracle-Database-related files. No changes were made at the OS level.

Table 7-2 shows the files and parameters edited for these tests.

Parameter	Description	Setting file
SDU (Session Data Unit)	Unit (byte) of data stored in a buffer during data transport. For tuning, a value of “32767” was set.	tnsnames.ora listener.ora sqlnet.ora
SEND_BUF_SIZE RECV_BUF_SIZE	Buffer size (byte) of TCP socket. For tuning, a value three times that of the bandwidth-delay product (BDP) was set. The BDP was calculated by multiplying the network bandwidth (bytes) by RTT (seconds).	tnsnames.ora listener.ora sqlnet.ora

Table 7-2: Network tuning

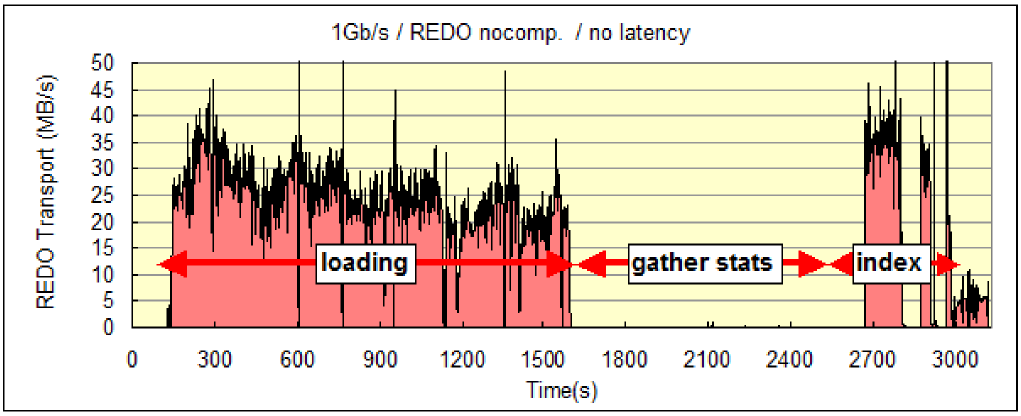
The settings were determined by referring to the following MAA white paper published by Oracle Corporation: Data Guard Redo Transport & Network Best Practices
http://www.oracle.com/technology/global/jp/products/availability/htdocs/maa.html

8. Verification Results

Discussed below are the findings and analysis of our verification tests.

8.1 Items to be considered in batch processing in an Oracle Data Guard environment

The following describes items to be considered in batch processing in an Oracle Data Guard environment. First, the amount of the redo to be transported during batch processing must be evaluated. Graph 8-1 shows the change in the volume of redo transported from the primary database to the standby database during various stages of processing. Arrows labeled “loading,” “gather stats,” and “index” on the graph indicate the execution of data loads, statistical data acquisition, and indexing, respectively. The bandwidth of the redo transport network was set to 1 Gb/s in this baseline test. The maximum volume of transported redo was approximately 40 MB/s, while the average value was about 15 MB/s. As indicated on the graph, redo was generated only during data loading and indexing. Virtually no redo was generated or transported during statistical data acquisition. The volume of transported redo indicates the total amount of the redo transported from the two nodes of the primary RAC database.

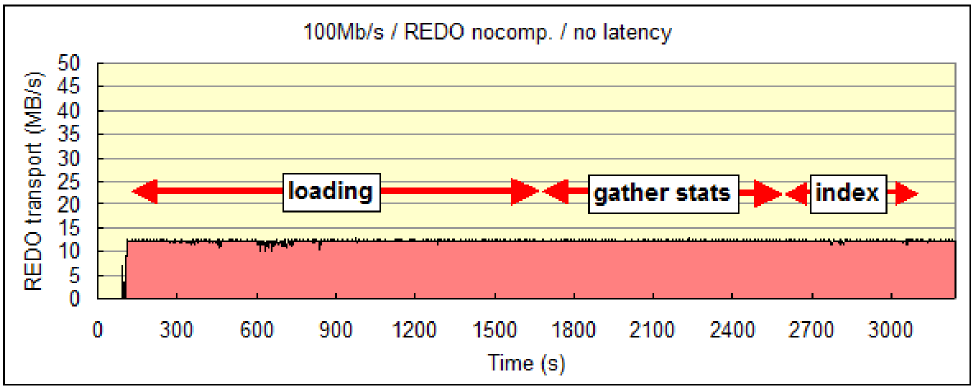


Graph 8-1: Volume of redo transported during batch processing (1 Gb/s)

A second test scenario ran the same batch processing workload but reduced the bandwidth available for Data Guard redo transport from 1Gb/s to 100Mb/s. Graph 8-2 shows redo transport volume for this second test pegged at 12.5 MB/s. The volume of redo being generated by batch processing far exceeds available bandwidth and thus consumes all of the available network bandwidth.[3] Also note that we continue to ship 12.5MB/s even during the phase where we are gathering stats. While gathering stats does not generate redo, the primary database uses that time to try and catch up on the backlog of redo

[3] 8 bits = 1 byte; therefore, 100 Mb/s = 12.5 MB/s

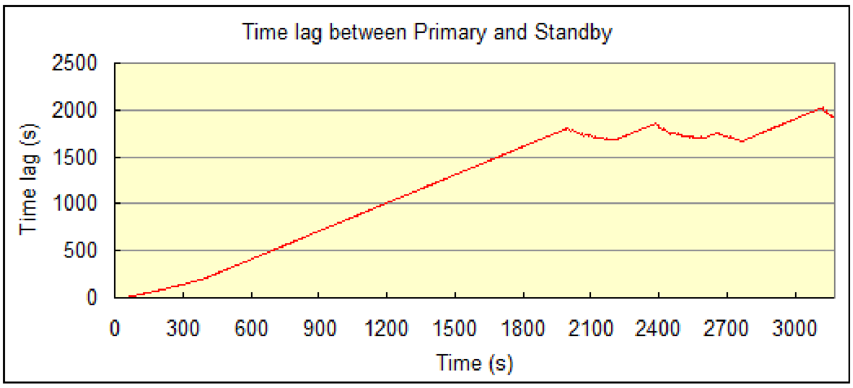
generated by the primary database but not yet shipped due to the bandwidth constraints.



Graph 8-2: Volume of redo transported during batch processing (100 Mb/s)

The inability of redo transport to keep up with updates of the primary database under these test conditions resulted in significant time lags between the primary and standby databases. Graph 8-3 shows the changes in time lag between the primary and standby databases under these test conditions. Here, *time lag* refers to the number of seconds by which the standby database state lagged behind the current primary database state, and is almost the same as the transport lag (redo not yet shipped) and apply lag (redo not yet applied) in our test environment. Time lag was measured by inserting a current timestamp at one-second intervals to the primary database and by searching for the newest timestamp at five-second intervals to the standby database – using the Active Data Guard option that allows the standby database to be open read-only while apply is active (alternatively you can use `V$DATAGUARD_STATS` to monitor transport and apply lag).

Graph 8-3 shows the time lag increasing over time. When attributed to a transport lag, such conditions can generate problems for data protection, affecting Data Guard's primary purpose.

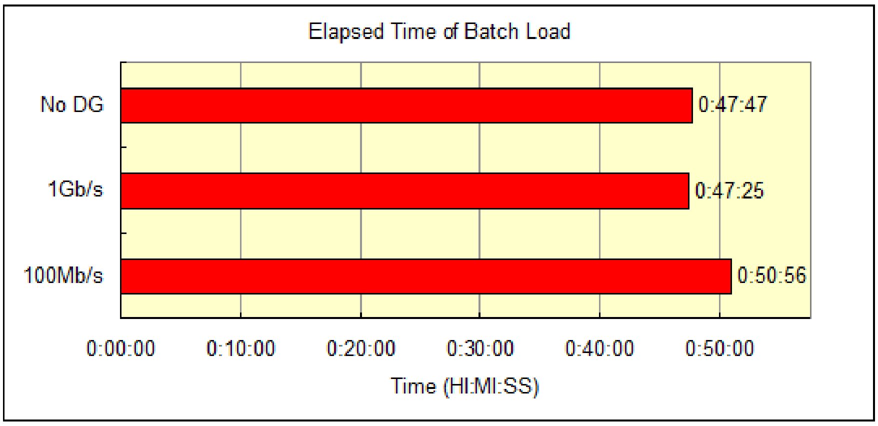


Graph 8-3: Changes in time lag between primary and standby databases during batch processing

The next series of tests compared batch processing times. Here, we examined three cases. In one case, redo transport was not initiated during batch processing (No DG). In the second case, redo was transported across a 1Gb/s network (1 Gb/s). In the third case, redo was transported across a 100Mb/s network (100 Mb/s). The results of this comparison are shown in Graph 8-4.

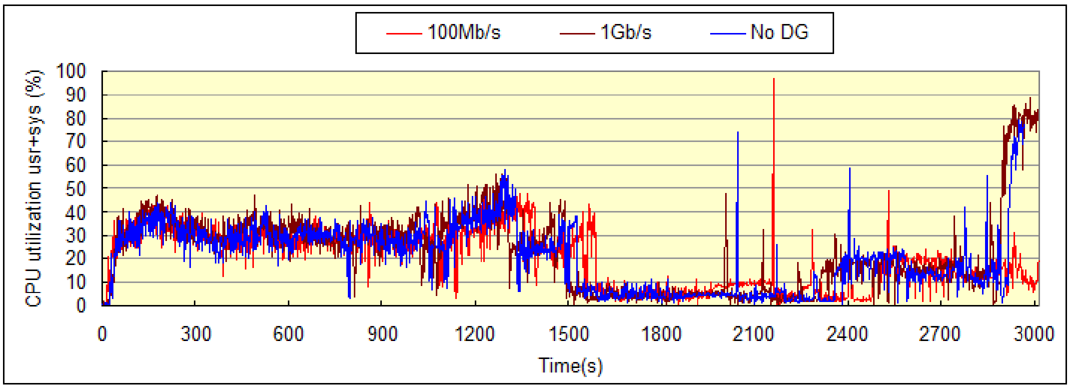
Batch processing times are nearly identical in the “No DG” case and “1 Gb/s” case, indicating that the redo transport does not affect batch processing performance.

In contrast, the “100 Mb/s” case has longer batch processing times than the “No DG” and “1 Gb/s” cases. This is explained by new functionality in Data Guard 11*g*. In the “1 Gb/s” case, redo transport is able to keep pace with redo generation and thereby benefit from a new capability of ASYNC to transmit redo directly from the redo buffer in SGA. In the “100 Mb/s” case, redo transport is unable to keep pace with generation, thus ASYNC transport is unable to empty the redo buffer before it must be recycled by the LGWR process, and ASYNC must revert to the Data Guard 10g functionality of reading from the primary database online log file. Thus the increased overhead observed in the 100MB/s case is due to increased I/O when ASYNC redo transport is forced to read from disk. Also note that the new Data Guard 11g functionality includes the ability for ASYNC redo transport to shift back and forth between reading from the redo buffer in SGA and reading from the online log file as required – without disrupting redo transmission.



Graph 8-4: Comparison of batch processing times

We also examined changes in CPU use during batch processing. Graph 8-5 shows the results of a CPU usage comparison, indicating no major difference in CPU usage in these cases. Although this test compares CPU usage using one node of the two-node RAC database server, CPU usage in the other node showed similar trends.



Graph 8-5: Comparison of CPU usage

Based on the results obtained in the preceding scenarios, we concluded the following items should be considered in cases where batch processing generates redo volume that exceeds the network bandwidth available to Data Guard:

(1) The potential for data loss may exceed your recovery point objectives.
(2) Batch processing performance may be affected.

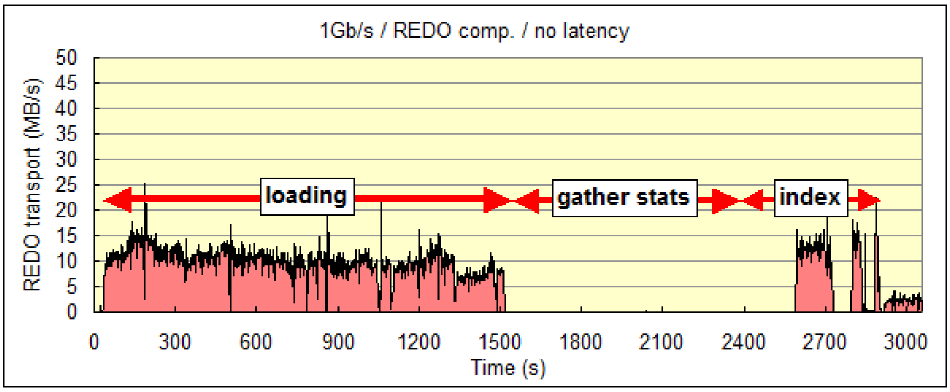
The phenomenon described in (1) is due to the fact that a transport lag will always occur when batch processing generates redo exceeding network bandwidth unless additional measures discussed later in this paper are taken. With regard to (2), the extent of the effect depends on batch processing specifics and configuration. The simplest way to solve these problems is to monitor the duration during which the volume of generated redo is highest in the target system and to provision network bandwidth that is greater than the redo generation rate. However, it can be difficult to ensure adequate network bandwidth in a system configuration in which the standby database is located far from the primary database for a disaster recovery.

Key point: logging and archive log mode

Oracle Data Guard requires the logging of all primary database processing in redo logs (logging) and enabling archiving (archive log mode) in order to function. Data Guard uses redo **-** the information used by the Oracle Database to recover transactions **-** to synchronize the standby database with the primary and thereby protect enterprise data. In certain workloads such as batch processing can generate large volumes of redo due to numerous updates, writing redo to redo logs and outputting archives can become a bottleneck. On our test environment, disks used as Online REDO Log were under high load and that affected batch processing performance. Eliminating this bottleneck requires examination of the disk configuration (use of high-speed disks or striping) to achieve faster redo log I/O.

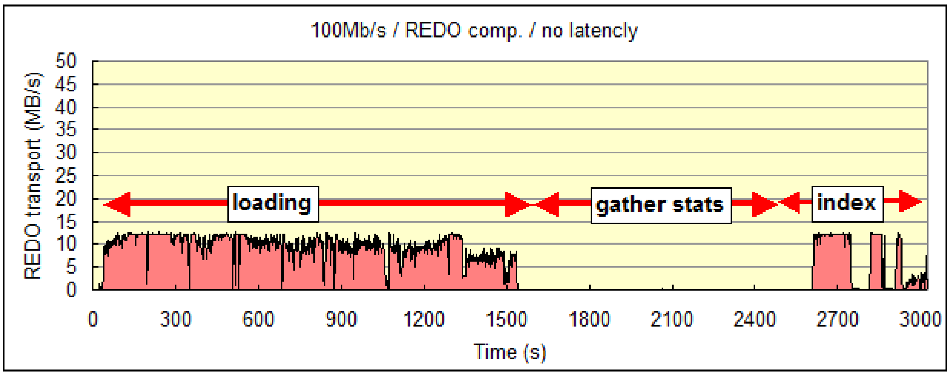
8.2 Effect of redo transport compression using ASYNC

This section discusses the effects of redo compression in batch processing when using Data Guard ASYNC redo transport. First, using a network bandwidth of 1 Gb/s, we checked the volume of redo transported in batch processing with asynchronous redo compression enabled[4]. Graph 8-6 shows the results of this test. By comparing these results to those shown in Graph 8-1, we confirmed that asynchronous redo compression reduced the volume of transported redo by more than half.



Graph 8-6: Volume of redo transported during batch processing (1 Gb/s)

With asynchronous redo compression enabled, the maximum volume of transported redo was approximately 15 MB/s. However, since the volume of transported redo exceeded the available bandwidth of 12.5 MB/s only briefly, changing the redo transport network bandwidth to 100 Mb/s would make no major difference in network usage. Graph 8-7 shows the result of this test.



Graph 8-7: Volume of redo transported during batch processing (100 Mb/s)

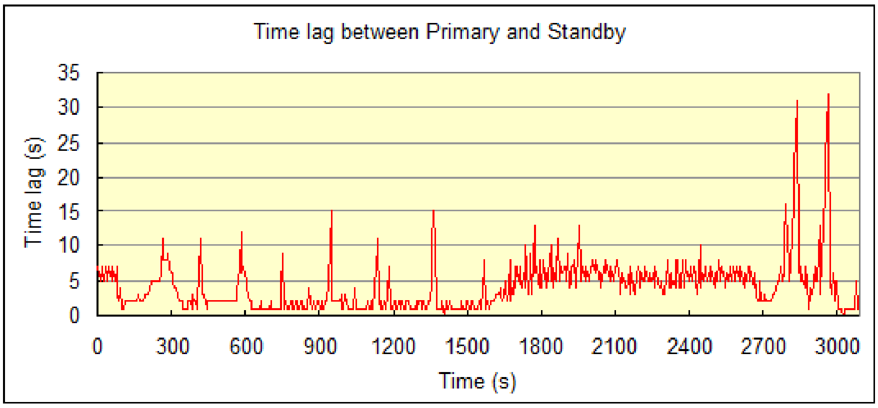
Notice how different the result in Graph 8-7 is from the uncompressed result in Graph 8-2. Redo transport compression has reduced the network volume and thus eliminated

---

[4] For information on setting the asynchronous redo compression, refer to MetaLink note: 729551.1.

bandwidth as a limiting factor. Redo transport is able to keep pace with redo generation. Further evidence of this is the fact that there is no longer any backlog of redo that needs to be transported during the 'gather stats' phase of processing when no new redo is being generated.

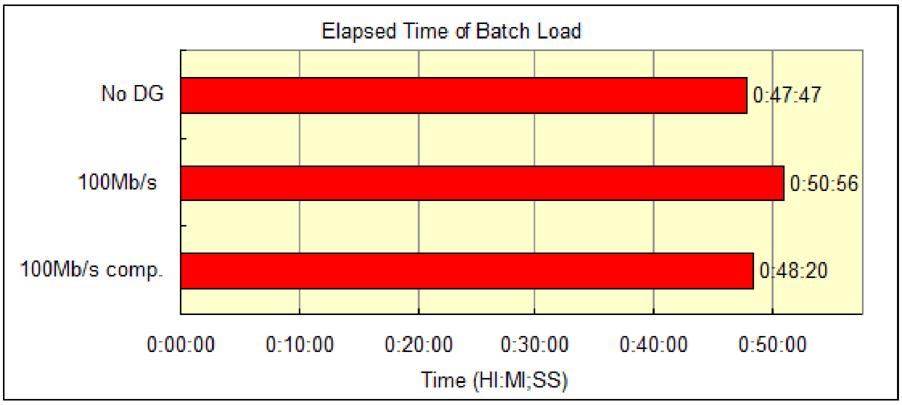
Graph 8-8 shows the time lag between the primary and standby databases measured in the test above. As indicated by this graph, no significant time lag was generated, and the standby database was able to follow changes in the primary database without delays. Compare this result to the significant time lag reported in Graph 8-3 when compression was not enabled. The use of redo transport compression in this scenario is very effective, both for data protection, and for making the standby database useful for offloading read-only queries and reports from the primary database.



Graph 8-8: Time lag between primary and standby databases during batch processing (with compression)

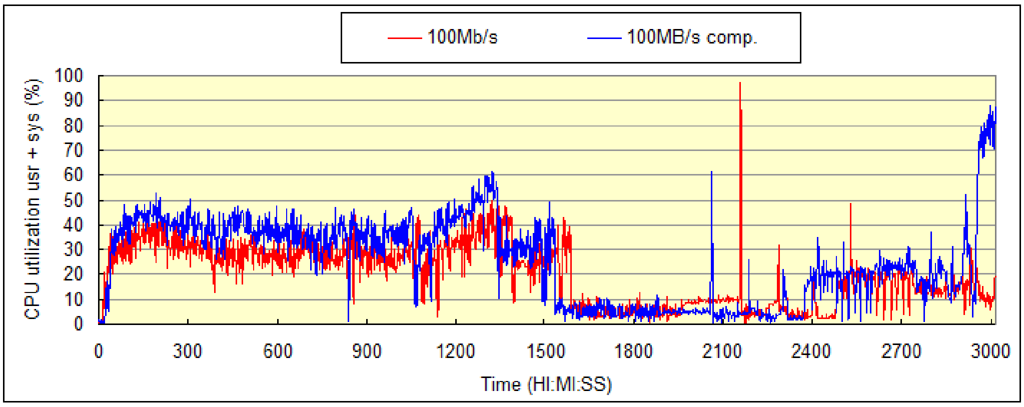
Next, we compared batch processing times using three cases. In one case, redo transport was not initiated during batch processing (No DG). In the second case, redo transport was enabled using a 100Mb/s network (100Mb/s). In the third case, asynchronous redo transport compression was enabled using a 100Mb/s network (100Mb/s comp.).

Graph 8-9 shows the results. Processing times were shorter in the "100 Mb/s comp." case than in the "100 Mb/s" case. There was no major increase in overhead when compared to the result in the "No DG" case. We concluded this result was achieved because compression prevented the volume of transported redo from exceeding the network bandwidth, enabling redo transport to ship directly from the log buffer and avoid disk I/O most of the time.



Graph 8-9: Comparison of batch processing times

Likewise, we compared the CPU usage in batch processing. Graph 8-10 shows the results. With asynchronous redo compression enabled, CPU usage was high at certain periods of redo transport, indicating that compression consumes CPU resources.



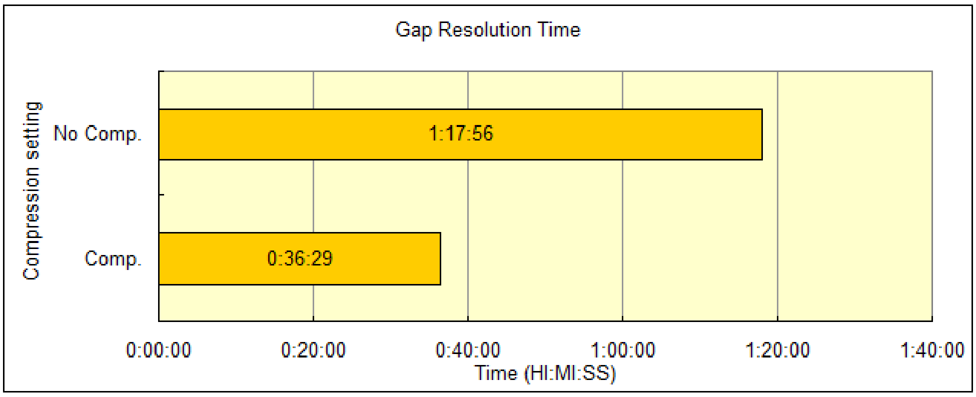
Graph 8-10: CPU usage during batch processing

As discussed above, asynchronous redo compression can reduce the volume of transported redo. Using our verification scenario, we confirmed that asynchronous redo compression can provide a significant benefit in bandwidth constrained environments by providing the required level of data protection and enabling the standby database to remain current with the primary. As long as there are available CPU resources to support compression, the asynchronous redo compression feature is highly effective in an Oracle Data Guard configuration.

8.3 Suspending redo transport during batch processing

It is also possible to suspend redo transport during batch processing in an Oracle Data Guard environment. This method offers the advantage of eliminating any potential impact of redo transport on batch processing performance; however, it also means that data resulting from batch processing is unprotected until the standby database is resynchronized at a later time. When we used this method in our verification, the redo generated during batch processing was transported by Data Guard's automatic gap resolution mechanism while OLTP processing was being executed after batch processing. We examined gap resolution time, the effects of the generated gap resolution process on OLTP processing, and the impact of gap redo compression.

First, we compare gap resolution times with and without gap redo compression. Here, gap resolution time refers to the time from start of OLTP processing (when redo transport was re-enabled) to the completion of application to the standby database of all redo generated by batch processing. The shorter this time period, the shorter the time period that data is unprotected, and the more quickly the data is available for queries and reports on the standby database. In this verification test, shorter gap resolution times enabled faster start of read-only processing using the standby database. Graph 8-11 shows the test results. As indicated by the graph, gap redo compression cut gap resolution time by more than half.

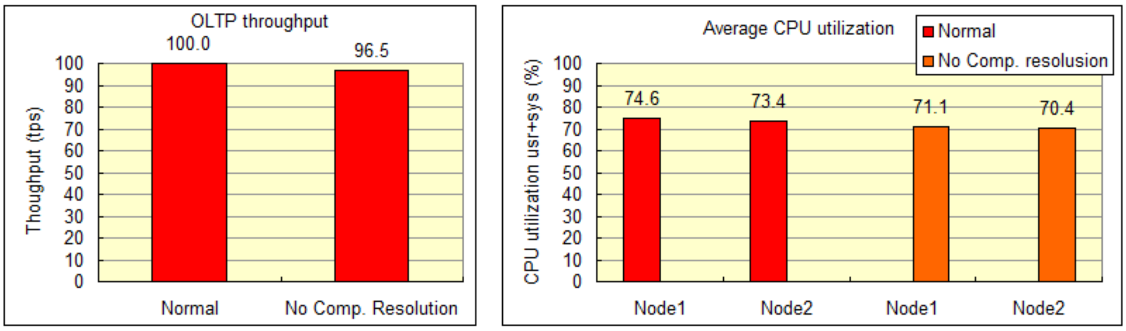


Graph 8-11: Comparison of gap resolution times

Next, we examined the impact of gap resolution on OLTP processing performance. Graph 8-12 compares average throughput and average CPU usage[5] between ordinary OLTP processing (Normal) and OLTP processing during gap resolution (No Comp. resolution). As indicated by the graph, both throughput and CPU usage were slightly lower in the "No Comp. resolution" case than in the "Normal" case, indicating that the slight drop in throughput is not due to the CPU. During gap resolution, both redo from the current log

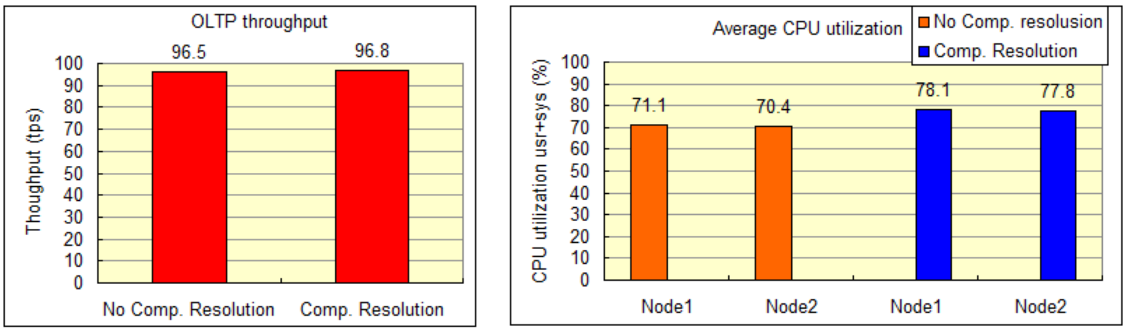
[5] We obtained average throughput and average TPS by calculating the average time between log switches. Average throughput values are coefficients calculated by assigning a reference value of "100" to the "Normal" case.

group is transported using Data Guard ASYNC transport while redo from the archived redo log are also transported using the Data Guard gap resolution process. This operation appears to have affected throughput slightly.



Graph 8-12: Comparison of OLTP processing performance (Normal vs. No Comp. resolution)

Graph 8-13 shows a comparison of OLTP processing in gap resolution in two cases, one with gap redo compression (Comp. resolution) and the other without (No Comp. resolution). With gap redo compression enabled, only the redo transported by gap resolution is compressed; the redo transported by OLTP processing is not compressed. The graph shows essentially no difference between the two cases. A careful look at CPU usage reveals about 7% CPU usage overhead at each node of the RAC database with gap redo compression enabled.



Graph 8-13: OLTP processing performance compression
(No Comp. resolution vs. Comp. resolution)

The volume of data processed by redo compression is determined by the volume of transported redo. For large volumes of transported redo, the volume of compressed redo also increases, increasing CPU usage overhead.

As mentioned at the start of this section, a method that suspends redo transport during batch processing eliminates the need to account for the impact of Oracle Data Guard on batch processing performance. Although this method provides this advantage, it also requires accounting for the following considerations:

(1) Deferring redo transport requires gap resolution after batch processing is complete, extending the length of time that data is unprotected and unavailable for read-only access at the standby database.
(2) Gap resolution operation does create additional overhead, however slight, that may impact the performance of processing that is occurring concurrent with gap resolution.

We confirmed the following based on results of verification testing carried out in accordance with the verification scenario.

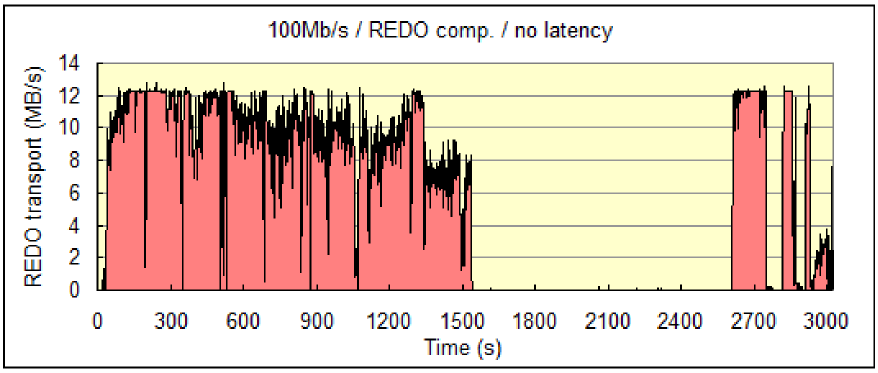
(1) Gap redo compression can reduce gap resolution time significantly in bandwidth constrained environments.
(2) Although gap resolution slightly reduced OLTP processing throughput, the effect was relatively negligible. Similar results were obtained with gap redo compression enabled as long as there was CPU available for compression processing.

8.4 Network tuning results

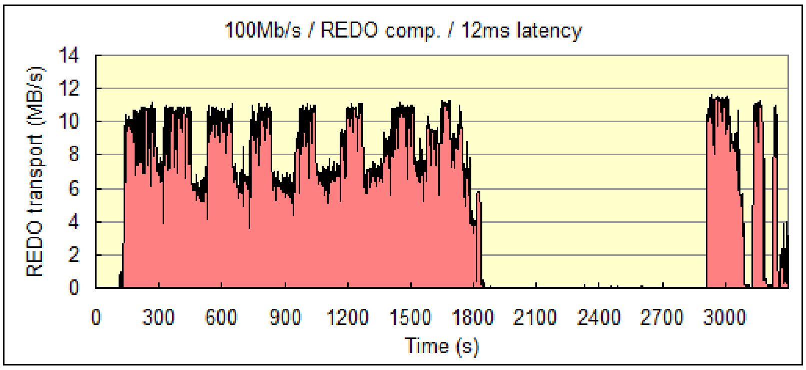
All verification tests described thus far were carried out without inducing RTT latency in the redo transport network. To simulate a more realistic disaster recovery system, we simulated additional latency in the redo transport network and ran the batch processing verification scenario. We tested scenarios with two different latency settings, one with 12-ms RTT latency and the other with a 40-ms RTT latency. Based on past case studies, the 12-ms latency assumes a communication distance equal to that between Tokyo and Osaka, while the 40-ms latency represents the communication distance between Tokyo and Okinawa. For verification tests, the redo transport network bandwidth was set to 100 Mb/s, with asynchronous redo transport compression enabled.

> **Note:** The relationship between transport distance and delay varies with network environments. The settings used in our verification tests should be regarded only as guidelines.

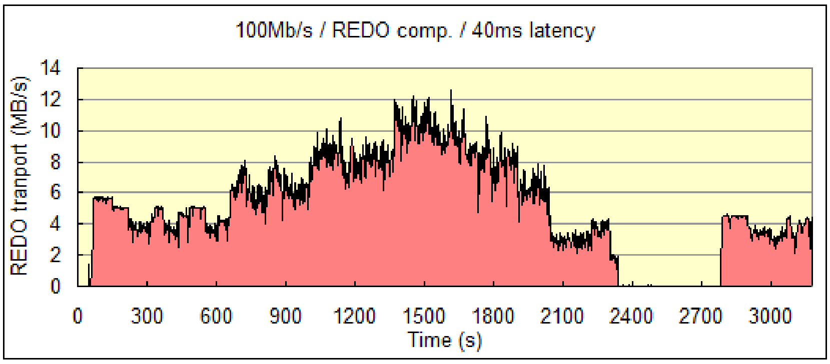
First, we confirmed the volume of transported redo in a number of scenarios in which redo transport was initiated during batch processing using default network settings. Graph 8-14 shows measurements without induced latency. Graph 8-15 shows the changes in the volume of transported redo with an RTT latency of 12 ms. Graph 8-16 shows measurements with an RTT latency of 40 ms. As the graphs show, increasing RTT latency impacts the volume of transported redo and reduces transport efficiency. This is an effect directly attributable to transport delay.



Graph 8-14: Change in volume of transported redo (without latency)



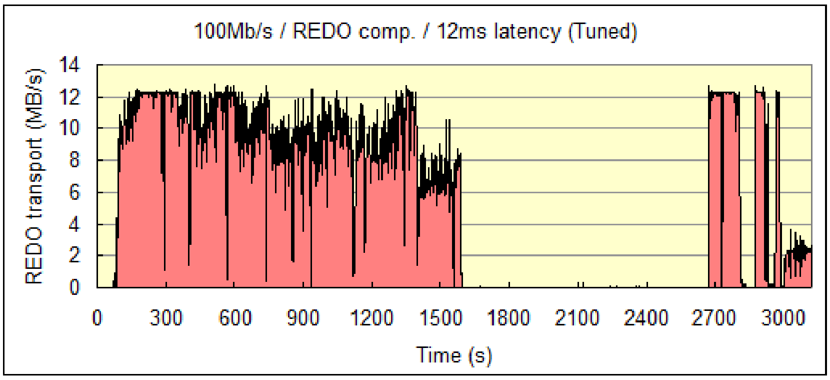
Graph 8-15: Change in volume of transported redo (12-ms latency)



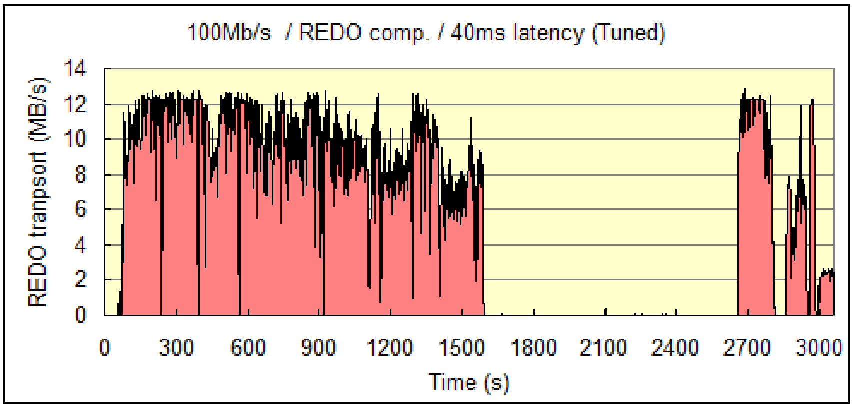
Graph 8-16: Change in volume of transported redo (40-ms latency)

In contrast, Graphs 8-17 and 8-18 show changes in the volume of transported redo after tuning described in the Maximum Availability Architecture white paper, Data Guard Redo Transport and Network Best Practices[6]. The graphs clearly indicate improvements in transport efficiency achieved after network tuning, resembling the graph in Graph 8-14 of redo transport without the delay setting.

6 http://www.oracle.com/technology/global/jp/products/availability/htdocs/maa.html

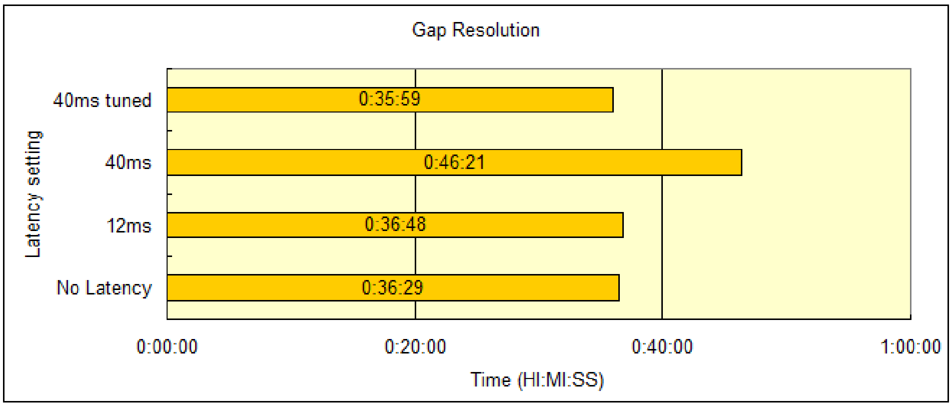


Graph 8-17: Change in volume of transported redo (12-ms latency tuned)



Graph 8-18: Change in volume of transported redo (40-ms latency, tuned)

Next, we examined cases in which redo transport was suspended during batch processing. Graph 8-19 shows the baseline results of testing using default network settings. The comparison of the result without latency and the result with a 12-ms latency indicates virtually no difference in gap resolution times. On the other hand, the comparison of results with no latency and with a 40-ms latency show longer gap resolution times. We hypothesized that this result was attributable to the lower transport efficiency of an environment that was not tuned to accommodate the increased network latency. Subsequent tuning confirmed this hypothesis, yielding gap resolution times at 40ms RTT latency that were similar to the test run with no latency, as shown in Graph 8-19.



Graph 8-19: Comparison of gap resolution times

These results confirm the effectiveness of network tuning. Transport efficiency using default settings is lower in cases where there is significant network latency, but proper network tuning significantly improves transport efficiency when using Data Guard 11g ASYNC redo transport.

9. Conclusion

This white paper focused on batch processing in a disaster recovery configuration. The verification tests were carried out with a disaster recovery environment configured using Hitachi BladeSymphony high-reliability blade servers, Oracle Database 11g, and Oracle Data Guard, and various batch processing scenarios designed to simulate operations under actual conditions. When designing batch processing in an Oracle Data Guard configuration, we need to consider processing performance (whether processing is completed within the batch window) and data protection (time lags between primary and standby databases). Our verification tests assumed two main scenarios: a scenario (DG) with redo transport enabled during batch processing and a scenario (No DG) in which redo transport is disabled during batch processing. We analyzed the effectiveness of redo transport compression (new in Oracle Database 11g) by enabling and disabling that feature (DG + Comp. and No DG + Comp.).

Table 9-1 below summarizes the characteristics of these different scenarios based on our verification test results with respect to batch processing performance and data protection.

Pattern	Batch processing performance	Data protection
DG	If the volume of transported redo exceeds the network bandwidth, overhead may result.	If the volume of transported redo exceeds the network bandwidth, time lags grow larger.
DG + Comp.	Compression generates virtually no impact to batch processing performance as long as there is available CPU for compression processing.	In a bandwidth-constrained environment, compression can reduce the volume of transported redo to a level such that the time lag does not increase, improving data protection and making the standby database more useful for read-only queries and reports.
No DG	Oracle Data Guard has no visible effects.	Update data is unprotected during batch processing. Gap resolution is required after batch processing. The standby database is not current with the primary database until gap resolution completes
No DG + Comp.	Oracle Data Guard has no visible effects.	Update data is unprotected during batch processing. Gap resolution is required after batch processing, but the resolution time or the network usage is much less than in the “No DG” case.

Table 9-1: Characteristics of batch processing in test scenarios

The results of this verification testing confirmed the efficacy of the redo compression feature in the Advanced Compression Option for Oracle Database 11g Enterprise Edition for bandwidth constrained environments and for systems having sufficient CPU capacity to support compression processing.

In actual applications, in addition to the characteristics listed in Table 9-1, related considerations for the allowable range of batch windows, gap resolution times, and CPU consumption for redo compression processing can be used to develop the optimal approach to batch processing in different Oracle Data Guard configurations.

We also confirmed that significant improvements in network throughput could be achieved in high RTT latency network environments (occurring when the standby database is located at a distance) by setting appropriate values for Oracle Database network parameters (`SDU`, `SEND_BUF_SIZE` and `RECV_BUF_SIZE`).

It is our hope that the data provided by these tests will prove useful for a wide range of implementations.

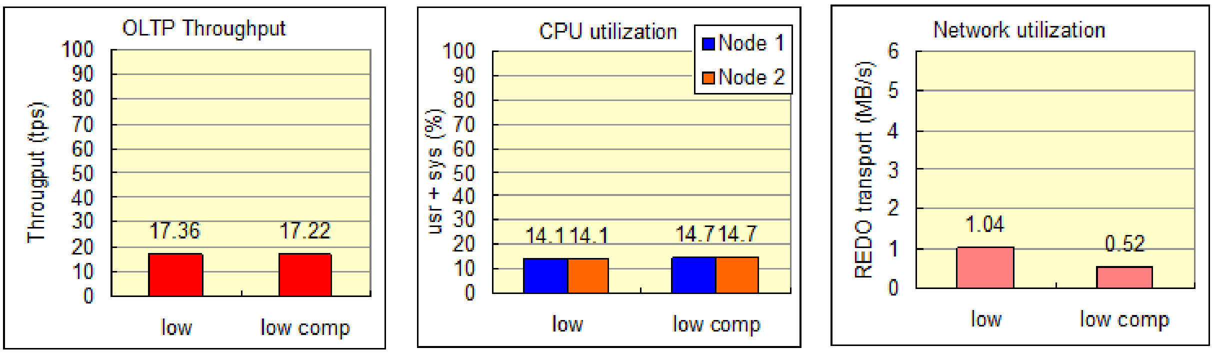
10. Acknowledgements

Oracle Corporation Japan formed a partnership with Hitachi Ltd. and other grid strategy partner companies in November 2006 to open the Oracle GRID Center (http://www.oracle.co.jp/solutions/grid_center/index.html), a facility that incorporates the most advanced technologies, with the goal of building next-generation business solutions capable of optimizing enterprise system infrastructures.

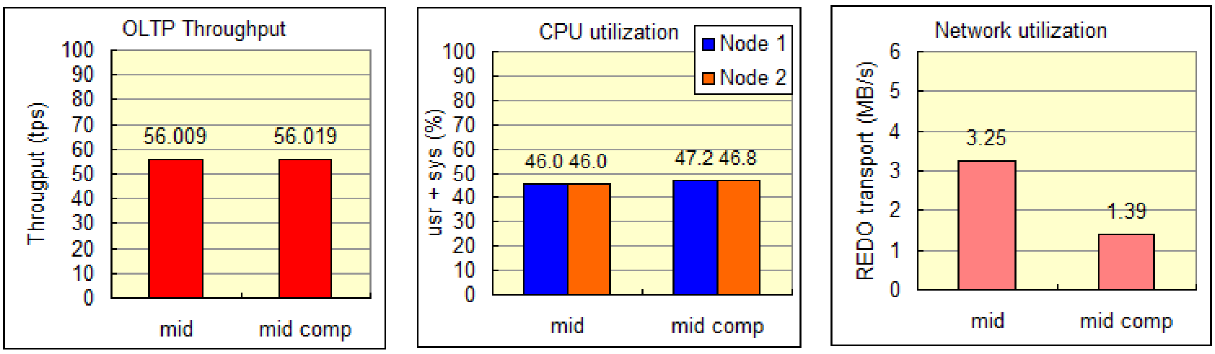
Publication of this white paper was made possible by hardware and software provided to the Oracle GRID Center by Intel Corporation and Cisco Systems G.K., which support the goals of the Oracle GRID Center, as well as support and aid provided by engineers from these companies. We wish to express our sincere gratitude to these companies and engineers for their support.

Appendix Effects of asynchronous redo compression in OLTP processing

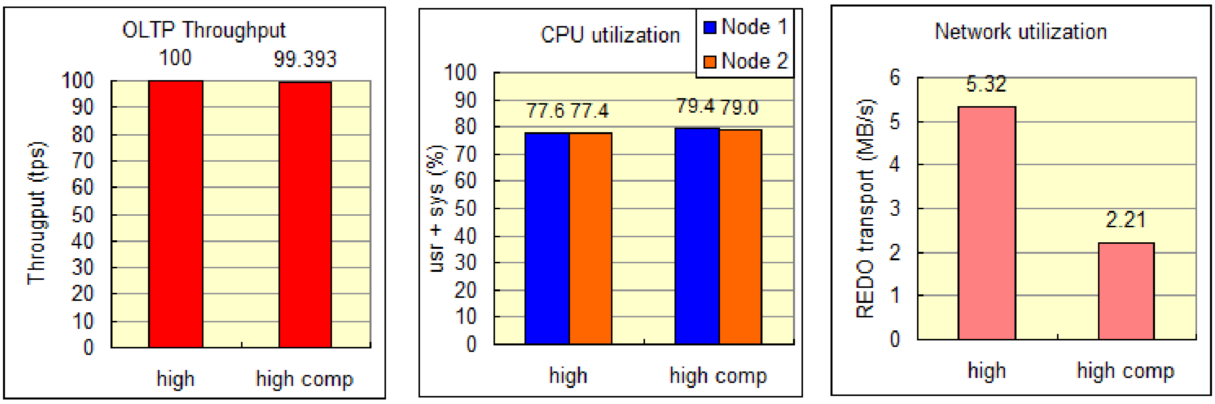
This white paper discusses the results of verification tests on batch processing in an Oracle Data Guard configuration. This Appendix provides data on the effects of asynchronous redo compression in OLTP processing. To analyze OLTP processing, we used a test application created for a hypothetical online shopping site, which was also used in the verification testing described in the main section of the white paper. In this verification test, performance measurements were obtained using three loads (low load (low), medium load (mid), and high load (high)). We then enabled asynchronous redo compression and reacquired performance measurements under the same load conditions (low load (low comp.), medium load (mid comp.), and high load (high comp.)). We compared performance measurements obtained with and without compression under the same load settings, then compared average throughput, CPU usage, and average network bandwidth usage. Average throughput values are coefficients calculated based on a reference value of “100” assigned to throughput under high load conditions without compression (high). The results of the verification test are shown in the graphs below.



Graph 1: Comparison of OLTP processing performance with redo compression enabled (low load)



Graph 2: Comparison of OLTP processing performance with redo compression enabled (medium load)



Graph 3: Comparison of OLTP processing performance with redo compression enabled (high load)

Under all load conditions, we found virtually no difference in throughput or CPU usage when using or not using asynchronous redo compression. On the other hand, enabling compression reduced average volumes of transported redo by half under low loads and to less than half under medium and high loads. This means asynchronous redo compression is highly effective in improving the efficiency of redo transport network bandwidth use in OLTP processing. Based on these results, we believe use of asynchronous redo compression with OLTP-intensive systems will readily minimize network-related costs.

Author

Yosuke Goto
Senior Engineer
Grid Center
Technology Readiness SC
Sales Consulting
Oracle Corporation Japan
yosuke.goto@oracle.com

Yoshihito Katayama
Engineer
Oracle Support & Solution Center
Platform Software Division
Hitachi Ltd.
yoshihito.katayama.bh@hitachi.com

Oracle Corporation Japan
Oracle Aoyama Center
2-5-8, Kita-Aoyama, Minato-ku, Tokyo
107-0061, Japan

Hitachi Ltd.
5030, Totsuka-cho, Totsuka-ku, Yokohama-shi,
Kanagawa 244-8555, Japan