11g Calibrate I/O Overview

Establishing a solid I/O subsystem is an essential part of building the infrastructure for an application. If any component in the I/O stack is limited in throughput, it will become the weakest link in the I/O flow. The I/O stack includes HBAs, storage switches, the storage array and physical disks. Oracle Corporation recommends that before an application is deployed and rolled into production, that the entire stack be verified to ensure it can meet the application service level agreements (SLA) and objectives (SLO).

It has always been a daunting task validating the I/O subsystem because real world workloads could not be easily reproduced. In Oracle Database 11g, the Real Application Testing feature (Capture/Replay) was introduced to inject real (captured) workload into the system. However, another new 11g feature is available to help assess the I/O capability of the database's storage system, and gauge maximum IOPS and Mbytes/s. This feature, which is included as part of the Database Resource Manager product, is called Calibrate I/O. This document will briefly review the Calibrate I/O feature and its usage.

The Calibrate I/O feature is based on a PL/SQL function called DBMS_RESOURCE_MANAGER.CALIBRATE_IO(). When Calibrate I/O is invoked it will generate I/O intensive read-only random I/O (db_block_size) and large-block (1MByte) sequential I/O workloads. Unlike various external I/O calibration tools, this tool uses the Oracle code stack and runs in the database, issuing I/O against blocks stored in the database. The results, therefore, much more closely match the actual database performance. Once the workload execution is completed, a summary of the results is provided.

The results from Calibrate I/O should be gauged against the expected throughput rate (the maximum overall throughput of the I/O subsystem). I/O calibration can be used to evaluate the performance of the storage subsystem and determine whether I/O performance problems stem from the database host or the storage subsystem.

The Oracle PL/SQL package DBMS_RESOURCE_MANAGER.CALIBRATE_IO is used to execute the calibration. The duration of the calibration is dictated by the NUM_DISKS variable as well as the number of nodes in the RAC cluster.

```
SET SERVEROUTPUT ON
DECLARE
  lat  INTEGER;
  iops INTEGER;
  mbps INTEGER;
BEGIN
--DBMS_RESOURCE_MANAGER.CALIBRATE_IO(<NUM_DISKS>, <MAX_LATENCY>,
iops, mbps, lat);
   DBMS_RESOURCE_MANAGER.CALIBRATE_IO (28, 10, iops, mbps, lat);

  DBMS_OUTPUT.PUT_LINE ('max_iops = ' || iops);
  DBMS_OUTPUT.PUT_LINE ('latency  = ' || lat);
  dbms_output.put_line('max_mbps = ' || mbps);
end;
/
```

Note that the first two variables (NUM_DISKS, MAX_LATENCY) are input variables, and the remaining three are output variables.

NUM_DISKS - To get the most accurate results, its best to provide the actual number of physical disks that are used for this database. The Storage Administrator can provide this value. Keep in mind that when ASM is used to manage the database files, say in the DATA diskgroup, then only physical disks that make up the DATA diskgroup should be used for the NUM_DISKS variable; i.e.; do not include the disks from the FRA diskgroup. In the example above the DATA diskgroup is made up of 28 physicals (presented as 4 LUNs or ASM disks)

LATENCY – This should be set to the defined response time SLA for your application; e.g., your 95th percentile response time SLA is 10secs.

Due to the high I/O load from running the Calibrate I/O workload, I/O calibration should only be performed when the database is idle, or during off-peak hours, to minimize the impact of the I/O workload on the normal database workload. The following are other considerations before invoking Calibrate I/O:

- Ensure asynchronous I/O is enabled on all datafiles and tempfiles. The following query can be used to verify asynchronous I/O for these files.

  ```
  col name format a50
  select name,asynch_io from v$datafile f,v$iostat_file i
  where f.file#=i.file_no
  and (filetype_name='Data File' or filetype_name='Temp File');
  ```

If asynchronous I/O is not enabled set disk_asynch_io=true. Note that on Linux, *async IO* can be silently disabled if the max number of async IO slots are used up. This can be one reason why the query above reflects *async off* when disk_asynch_io is true. The max number of async IO slots can be found in /proc/sys/fs/aio-max-nr and the currently used slots can be found in /proc/sys/fs/aio-nr

- The database where calibration is to be executed, needs to be quiesced or the calibration results may become "tainted". Additionally, if other databases are running on the servers, then the calibration execution may adversely affect their performance.

- For RAC enabled databases, ensure that all instances of the RAC cluster are started, this will ensure that a complete calibration is performed. The calibration will run across all the nodes that are active and part of the RAC cluster

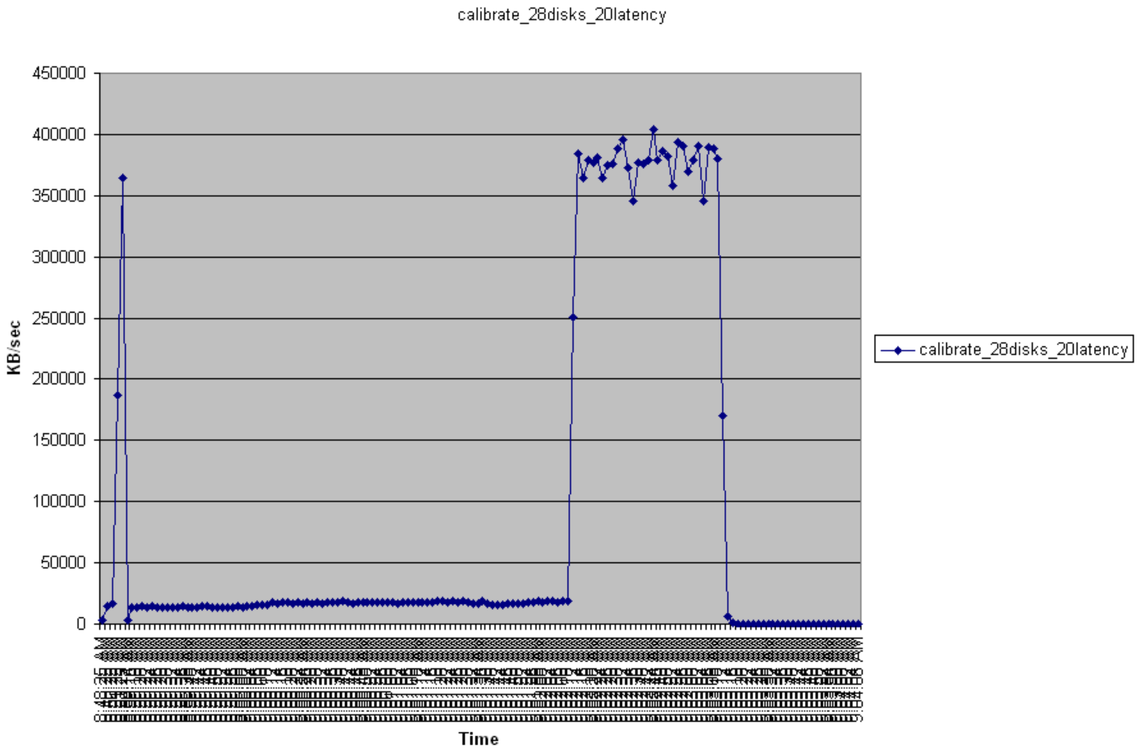
- Ensure that there's only one calibration execution active at a time and execute it only from one node; i.e.; do not execute multiple times across separate databases that use the same storage subsystem. Query the V$IO_CALIBRATION_STATUS view to see current calibration status

The calibration will run in different phases. In the first phase, small block random I/O workload is performed on each node and then concurrently on all nodes. The second phase will generate large block sequential I/O on each node. Note, that the Calibrate I/O is expecting that a datafile is spread across all disks specified in NUM_DISKS variable. Furthermore, offline files are not considered for file I/O.

Once the calibration is completed the results are shown immediately, as shown below, and also stored in the DBA_RSRC_IO_CALIBRATE table.

```
SQL> @calibrate
max_iops = 3100
latency  = 20
max_mbps = 376
```

Below is a chart showing the read throughput (in KB) during an execution of Calibrate I/O. This configuration includes a two-node RAC cluster with 2 4Gbyte FC HBA cards each. The HBAs are running in active/passive mode. As the chart shows, this two-node RAC cluster generated approximately 380Mbytes/s for the large block calibration, and then 3100 IOPS for the small block random I/O calibration.



Although the IOPS numbers look quite reasonable, the 376 Mbytes/s is much lower than expected. Consequently, CalibrateIO exposed areas, and indicated a need for capacity planning and/or performance management activities to bring the Mbytes/s rate closer to expected rate.

Additionally, the V$IOSTAT_FILE view has current information on I/O statistics.

```
SQL> select
FILE_NO,SMALL_READ_MEGABYTES,SMALL_READ_REQS,LARGE_READ_MEGABYTES,LARGE_READ_RE
QS from v$iostat_file

FILE_NO  SMALL_READ_MEGABYTES  SMALL_READ_REQS   LARGE_READ_MEGABYTE  LARGE_READ_REQS
-------  --------------------  ----------------  -------------------  ---------------
      5                    33              4167                  181              181

      6                     4               523                   19               19

      7                  2557            327319                14620            14617
```

Conclusion

The new IORM and CalibrateIO features are valuable tools in understanding limitations of the current I/O architecture. Once the calibration is completed, information is available to perform appropriate I/O design and sizing.

White Paper Title
July 2008
Author: Nitin Vengurlekar
Contributing Authors: Michael Nowak

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
oracle.com