Reclaim Oracle® Database Capacity with Nimble Storage on Oracle Linux

How to Use Oracle ASRU Tool with Nimble Storage

ORACLE WHITE PAPER | JULY 2014

Disclaimer

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Overview

Today's database administrators and architects understand the importance of storage to maintain quality of service and operate within budget. Storage becomes more essential as organizations grow and place greater demands on their IT infrastructure.

Traditional methods like adding disks to increase performance have not been effective. A more modern approach is to select a storage system that can meet the capacity demands of the business-critical applications supported by Oracle Databases.

Thin provisioning has been the most popular answer to the capacity issue. However, with thin provisioning, the greatest capacity savings only occur initially then erode over time because most storage systems are not interoperable with operations common to Oracle Databases. Nimble Storage Thin Provisioning automatically detects Oracle operations eliminating space erosion.

This technical paper demonstrates how database administrators can capitalize – without additional effort – on the capacity savings achieved by Nimble Storage Thin Provisioning as well as to maintain those savings over time. And it will detail how Oracle Automatic Storage Management (ASM) works with Nimble Storage solutions to claim unused (but allocated) disk space automatically and non-disruptively.

The steps outlined in this paper are based on a system running Oracle Linux with Unbreakable Enterprise Kernel. This environment was chosen because it suites both enterprise and development environments; Deep testing and integration between the layers on Oracle Linux for Oracle Database workloads brings substantial benefits to production environments (fast transaction speeds, scalable performance, security and reliability), while low cost and ease of maintenance make it ideal for development environments (Oracle's public yum server makes all errata and security patches freely available to users without any subscription or license fees).

Background

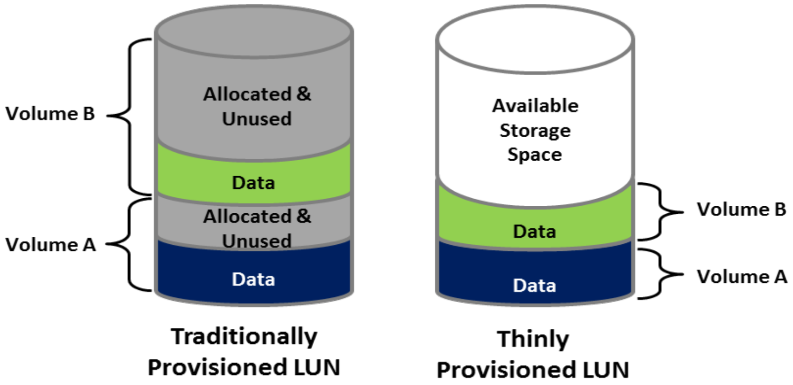
Storage cost is a growing concern for most enterprises, both in terms of capital expenses (CAPEX) and administrative overhead (OPEX). ASM is a purpose-built file system and volume manager integrated into Oracle Databases. Oracle ASM dramatically reduces administrative overhead (OPEX) associated with managing storage. Nimble Storage Thin Provisioning, available at no additional cost, dramatically reduces acquisition costs (CAPEX).

Thin provisioning is an efficient approach to provisioning storage. The storage array allocates capacity to the thinly provisioned storage volume as data is written, reducing the cost associated with provisioning upfront capacity at the time when storage volumes are created. Most enterprise applications on traditional arrays utilize only 20 to 45 percent of allocated storage, leaving more than half the disk capacity (and half the CAPEX) unavailable to other applications. With thin provisioning, nearly 100 percent utilization of available storage can be achieved. As a result, IT can significantly reduce the amount of storage needed to support Oracle Databases. In other words, thin provisioning allows organizations to defer storage purchases and defer capital expenditures, but it is important to note that the administrative overhead required to maximize utilization with thin provisioning differs among storage arrays.

Oracle Database 11g Release 2 and Oracle Database 12c Release 1 support Nimble Storage Thin Provisioning with a database feature known as “autoextend.” Autoextend is to databases what thin provisioning is to storage systems, allowing file size for tablespace to grow as applications add new data to database tables.

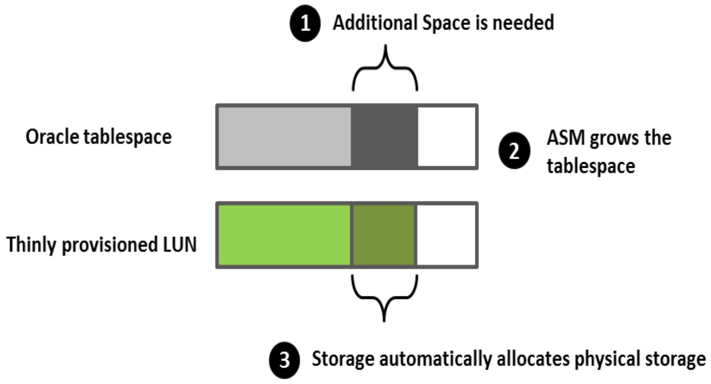
Thin Provisioning

Thin provisioning allows for on-demand allocation rather than up-front allocation of physical storage. Storage arrays present storage devices to the operating system as Logical Unit Numbers (LUNs) measured in physical blocks of 512 bytes. Historically the storage array reserved the precise number of physical blocks on disk specified as the LUN size. The allocation of physical blocks on LUNs presented to the operating system occurred when the storage administrator created it, and over time data is written to the LUN until the LUN fills up.



With thin provisioning, allocation occurs only when blocks are written to the array during use. When a thinly provisioned volume is created, the storage array presents the operating system with a LUN of the specified size. The actual physical size of the storage supporting the LUN is much smaller, about equal to the actual data written to the LUN. As applications write to a LUN, the storage array automatically and transparently grows the LUN in the background until it reaches maximum allocated size. Thin provisioning eliminates the need to reserve disk space for every LUN.

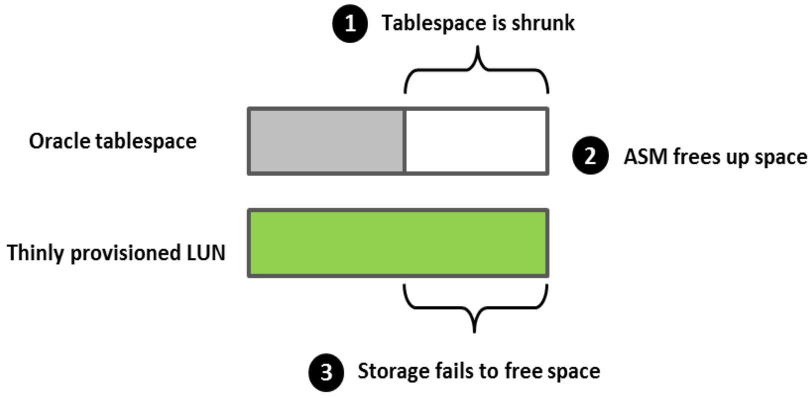
An Oracle ASM disk group can be created using several thinly provisioned LUNs. In this scenario, the database is created in the disk group and its primary tablespace will leverage the Oracle Database Autoextend feature. As additional space is needed, the database tablespace automatically grows. In a test performed by Oracle and Nimble Storage Technical Marketing Engineers, the file backing the tablespace resides in an Oracle ASM disk group. When the database writes to newly allocated database blocks during an autoextend action, the storage array automatically allocated physical storage to support the tablespace growth.


1 Additional Space is needed
Oracle tablespace
2 ASM grows the tablespace
Thinly provisioned LUN
3 Storage automatically allocates physical storage

Efficiency Erosion in Thin Provisioned Storage

Thin provisioning, alone, cannot free space that has been de-allocated or otherwise deleted at the application level. Over time this leads to erosion of the efficiency that thin provisioning provides when the storage array is initially deployed. To help with this issue Oracle introduced a new capability to ASM that extends support for thin provisioning. Previously, the amount of storage allocated to a tablespace would increase as the tablespace became larger. If a tablespace shrank – or if an entire database were removed from an Oracle ASM disk group – capacity previously allocated inside the array could not be reclaimed and reallocated to another application. The new ASM capability, ASRU (ASM Storage Reclamation Utility), allows a storage array to detect unused space freed by ASM and return it to unallocated status within the array.

However, traditional storage arrays natively cannot detect when disk space used by a deleted tablespace has been freed for de-allocation by ASRU. The space remains allocated inside the storage array.

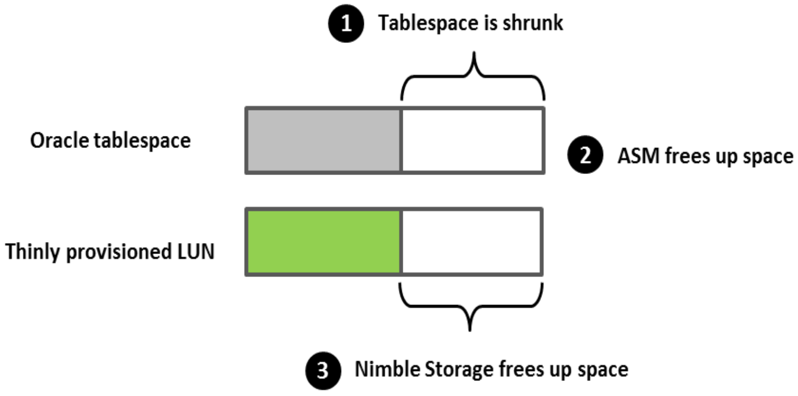


This problem occurs across most application sets. The storage industry addresses this shortcoming by extending the SCSI standard to incorporate a reclaim operation. In this operation, a new SCSI command tells the array which regions on a LUN have been de-allocated after an application deletes file space so that they become available for re-use.

This requires operating system support at the device driver and file system level, as well as from the storage array platform. Most storage vendors endorse the idea of using SCSI commands to reclaim capacity as a possible answer to space reclamation.

A Better approach to Eliminate Erosion

Nimble Storage chose an alternative approach to reduce or eliminate erosion of storage efficiency called “zero detection." Zero detection is useful for operating systems that do not, or have yet to, support the SCSI reclamation command. With zero detection, Nimble Storage firmware automatically classifies blocks written with zeros as de-allocated. As a result, database administrators do not have to run SCSI reclamation on storage systems that support zero detection to de-allocate blocks after the tablespace shrinks or an entire database is removed.



While Oracle Linux also supports the SCSI reclamation commands required by other storage solutions, below is an example of the simplified approach Nimble Storage uses to prevent space erosion using Oracle Database 12c Release 1, and Oracle Linux 6 Update 4 with Unbreakable Enterprise Kernel as an example environment:

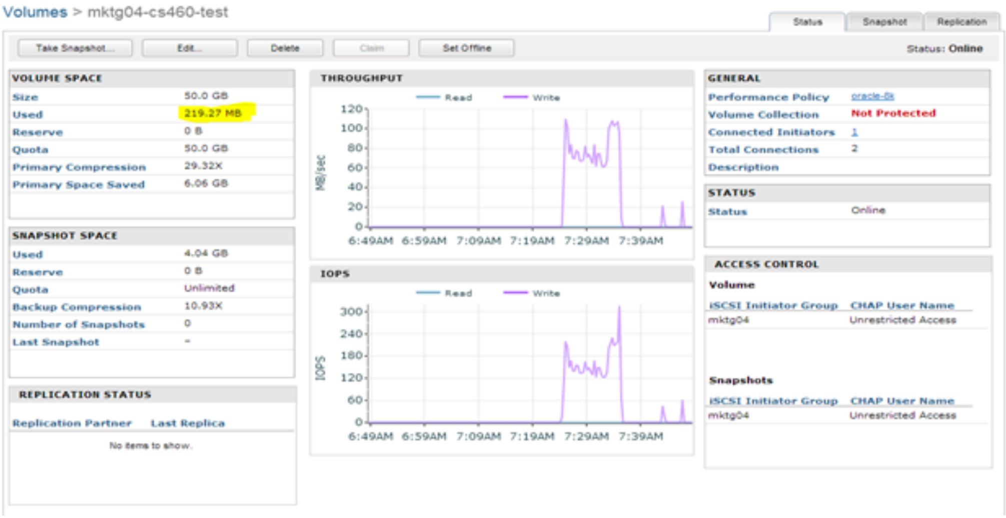
Pre-work:

1. Create a thinly provisioned LUN and map it to a database server.
2. Create a disk group on the thin provisioned volume.
    - Name it “TESTDG”
3. Create a tablespace on the disk group

- Name it “TESTTS”

Steps taken:

1. Verify initial volume size of a thinly provisioned LUN



Status: **Online**

VOLUME SPACE	
Size	50.0 GB
Used	219.27 MB
Reserve	0 B
Quota	50.0 GB
Primary Compression	29.32X
Primary Space Saved	6.06 GB

SNAPSHOT SPACE	
Used	4.04 GB
Reserve	0 B
Quota	Unlimited
Backup Compression	10.93X
Number of Snapshots	0
Last Snapshot	-

REPLICATION STATUS

Replication Partner	Last Replica

No items to show.

GENERAL	
Performance Policy	oracle-8k
Volume Collection	Not Protected
Connected Initiators	1
Total Connections	2
Description	

STATUS	
Status	Online

ACCESS CONTROL

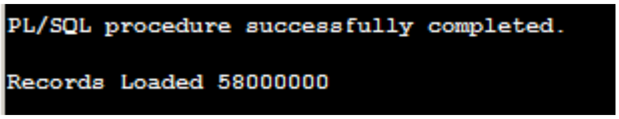
Volume

iSCSI Initiator Group	CHAP User Name
mktg04	Unrestricted Access

Snapshots

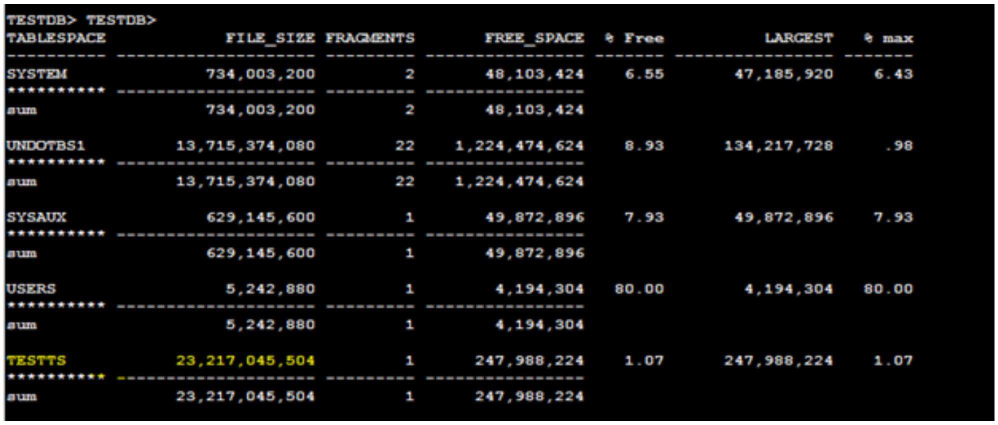
iSCSI Initiator Group	CHAP User Name
mktg04	Unrestricted Access

2. Insert 58,000,000 records in a table



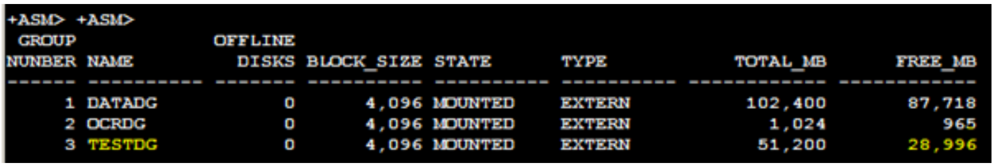
```
PL/SQL procedure successfully completed.

Records Loaded 58000000
```

3. Check the tablespace size



```
TESTDB> TESTDB>
TABLESPACE           FILE_SIZE FRAGMENTS         FREE_SPACE  % Free           LARGEST   % max
---------- -------------------- --------- ------------------ ------- ----------------- -------
SYSTEM             734,003,200         2         48,103,424    6.55        47,185,920    6.43
********** -------------------- --------- ------------------
sum                734,003,200         2         48,103,424

UNDOTBS1        13,715,374,080        22      1,224,474,624    8.93       134,217,728     .98
********** -------------------- --------- ------------------
sum             13,715,374,080        22      1,224,474,624

SYSAUX             629,145,600         1         49,872,896    7.93        49,872,896    7.93
********** -------------------- --------- ------------------
sum                629,145,600         1         49,872,896

USERS                5,242,880         1          4,194,304   80.00         4,194,304   80.00
********** -------------------- --------- ------------------
sum                  5,242,880         1          4,194,304

TESTTS          23,217,045,504         1        247,988,224    1.07       247,988,224    1.07
********** -------------------- --------- ------------------
sum             23,217,045,504         1        247,988,224
```

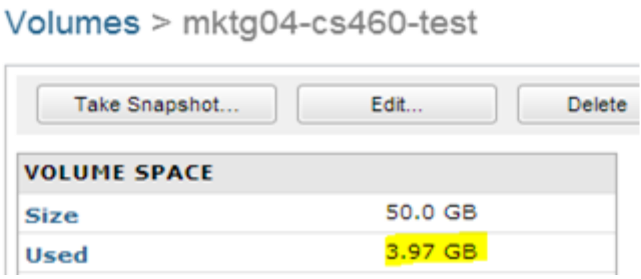
4. Check the disk group size



```
+ASM> +ASM>
 GROUP                OFFLINE
NUMBER NAME             DISKS BLOCK_SIZE STATE      TYPE          TOTAL_MB      FREE_MB
------ ---------- ------- ---------- ---------- ---------- ------------ ------------
     1 DATADG               0      4,096 MOUNTED    EXTERN         102,400       87,718
     2 OCRDG                0      4,096 MOUNTED    EXTERN           1,024          965
     3 TESTDG               0      4,096 MOUNTED    EXTERN          51,200       28,996
```

5. Check the physical LUN size

   5.1. With compression



Volumes > mktg04-cs460-test

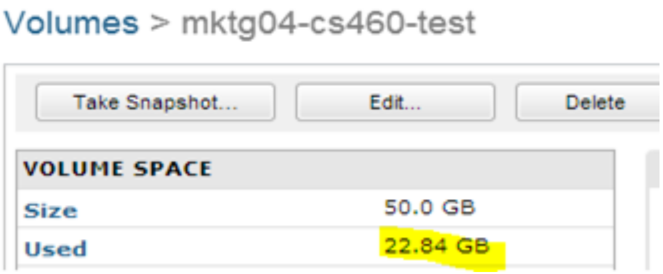
VOLUME SPACE

Size 50.0 GB

Used 3.97 GB

Observation: The reason you see only 3.97 GB used on the LUN vs. 22.84 GB on the disk group is because of compression on the Nimble Storage.

   5.2. Without compression



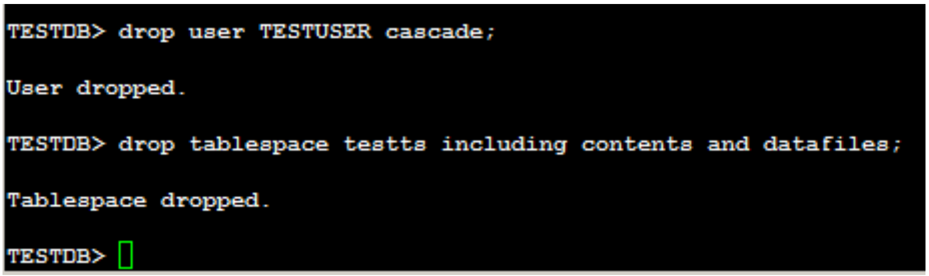
Volumes > mktg04-cs460-test

VOLUME SPACE

Size 50.0 GB

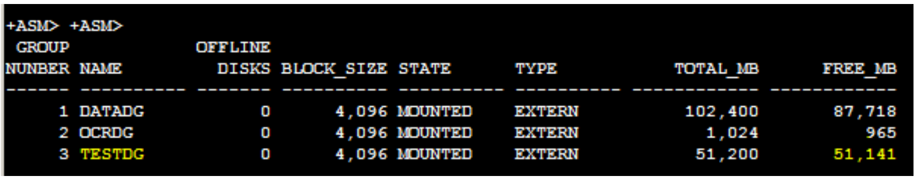
Used 22.84 GB

6. Drop tablespace TESTTS



```
TESTDB> drop user TESTUSER cascade;

User dropped.

TESTDB> drop tablespace testts including contents and datafiles;

Tablespace dropped.

TESTDB>
```

7. Check ASM disk group size after drop

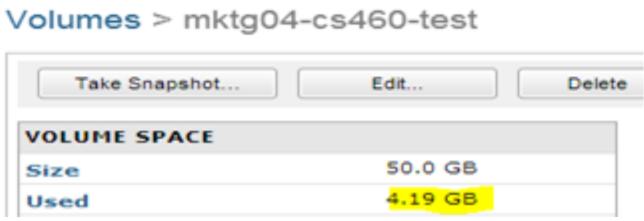


```
+ASM> +ASM>
 GROUP                OFFLINE
NUMBER NAME             DISKS BLOCK_SIZE STATE      TYPE             TOTAL_MB      FREE_MB
------ ---------- ------- ---------- ---------- ---------- ------------ ------------
     1 DATADG               0      4,096 MOUNTED    EXTERN            102,400       87,718
     2 OCRDG                0      4,096 MOUNTED    EXTERN              1,024          965
     3 TESTDG               0      4,096 MOUNTED    EXTERN             51,200       51,141
```

Observation: Notice that the disk group size has been freed up, but the physical LUN size has not. Also the used disk space increased slightly because of compression without data. When the tablespace was deleted, the Oracle ASM extents no longer contain data.

8. Check volume size after the tablespace was deleted

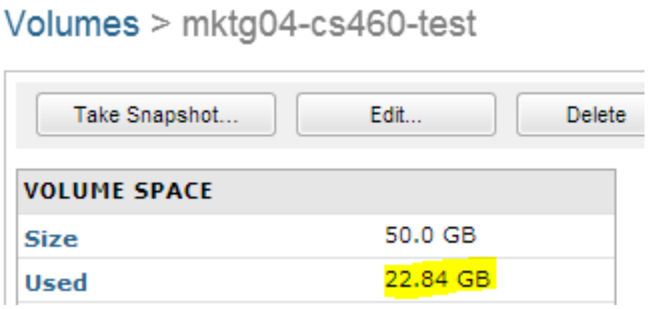
   8.1. With compression



Volumes > mktg04-cs460-test

VOLUME SPACE	
Size	50.0 GB
Used	4.19 GB

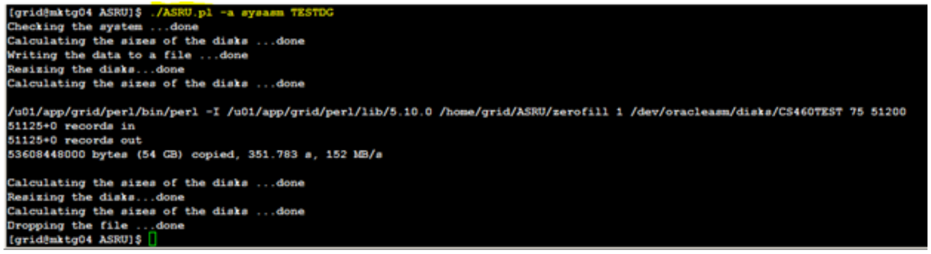
   8.2. Without compression



Volumes > mktg04-cs460-test

VOLUME SPACE	
Size	50.0 GB
Used	22.84 GB

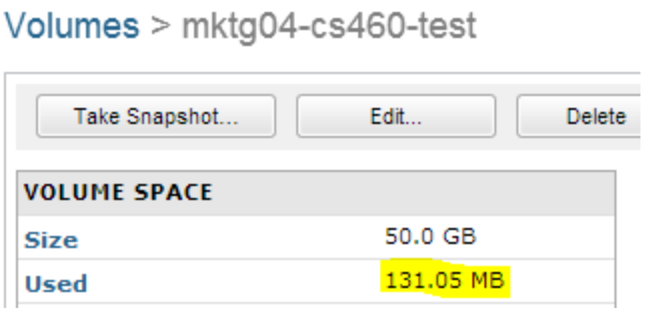
9. Execute the ASRU command



```
[grid@mktg04 ASRU]$ ./ASRU.pl -a sysasm TESTDG
Checking the system ...done
Calculating the sizes of the disks ...done
Writing the data to a file ...done
Resizing the disks...done
Calculating the sizes of the disks ...done

/u01/app/grid/perl/bin/perl -I /u01/app/grid/perl/lib/5.10.0 /home/grid/ASRU/zerofill 1 /dev/oracleasm/disks/CS460TEST 75 51200
51125+0 records in
51125+0 records out
53608448000 bytes (54 GB) copied, 351.783 s, 152 MB/s

Calculating the sizes of the disks ...done
Resizing the disks...done
Calculating the sizes of the disks ...done
Dropping the file ...done
[grid@mktg04 ASRU]$ 
```

10. After the ASRU operation completed, the physical LUN space is freed up.

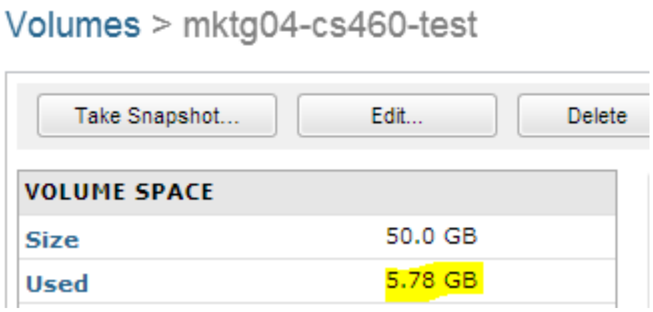
    10.1. With compression



Volumes > mktg04-cs460-test

VOLUME SPACE	
Size	50.0 GB
Used	131.05 MB

10.2. Without compression



Volumes > mktg04-cs460-test

VOLUME SPACE	
Size	50.0 GB
Used	5.78 GB

Conclusion: Nimble Storage not only allows the DBA to reclaim space on the physical LUN, but also enables him to achieve a very good compression rate on the LUN. The TESTDG disk group space usage after 58,000,000 records were loaded in a table was only 4 gigabytes (GBs) as opposed to 22GBs of space with the physical LUN.

Oracle ASRU Utility: A Better approach to De-allocation

A stand-alone utility (a PERL script) is used to reclaim previously allocated storage in an ASM disk. The utility, *ASRU (**A**SM **S**torage **R**eclamation **U**tility)*, accepts the name of the disk group for which space should be reclaimed. It writes blocks of zeros to regions on ASM disks where space is unallocated. The Nimble Storage array detects the writing of zero blocks and frees up space.

Three phases of an Oracle ASRU script working with Nimble Storage:

Compaction. This first phase, in which ASRU logically resizes the disks downward.

The amount of space in the disk group is resized to equal that of the file space (plus a small amount for reserved capacity). By default the value for reserved capacity is 25 percent, which can be changed manually in this utility. The resize operation is logical (built-in) to ASM and does not evoke any change on the physical disks. File data in the ASM disk group is compressed near the disks' beginning. The utility uses the appropriate database V$ table to determine the current allocated size of the disk group. The ASM Power Limit system setting controls the re-balance rate.

De-allocation. The second phase, in which ASRU writes zeros to the region where the ASM disks have been resized.

This phase begins only after the ASM rebalance of the disk group is verified as complete. After which, the ASRU script calls upon the subscript, zerofill, to write zeros to unused areas on the disks. This is when Nimble Storage automatically classifies blocks written with zeros as de-allocated.

Expansion. In the final phase, all the ASM disks are returned to the size determined when ASRU script was initiated.

This operation is a logical resizing of the disks and does not reorganize the file data. Once this operation is complete, most of the unallocated storage being has been freed.

Conclusion

As capacity needs have grown, thin provisioning has become an important mechanism for enterprise storage solutions. Thin provisioning not only reduces organizational storage requirements, it lightens administrative overhead associated with storage provisioning.

Nimble Storage Thin Provisioning is native, easy to use, and available at no additional cost. It works with Oracle ASM Storage Reclamation to offer Oracle Database users greater control over capital expenses in what has become an increasingly dynamic database environment.

When you deploy Oracle Database on Oracle Linux with Nimble Storage, you have the confidence of deploying on a platform backed by development teams that work closely together to optimize performance, enterprise security, and availability while delivering solutions that are easy and more cost efficient to administer.

Resources:

Oracle products:

Oracle Linux

Oracle Virtualization

Oracle ASRU Whitepaper

Nimble Storage:

http://www.nimblestorage.com/solutions/business-applications/oracle.php

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

Worldwide Inquiries
Phone: +1.650.506.7000
Fax: +1.650.506.7200

Hardware and Software, Engineered to Work Together

Authors: Ibrahim "Ibby" Rahmani & Tom Dau, Nimble Storage, Phillip Goerl, Oracle.