White Paper

Implementing Virtual Provisioning on EMC Symmetrix DMX with Oracle 10g and Oracle 11g

Applied Technology

Abstract

This white paper provides a detailed review of the technical aspects and benefits of deploying Oracle 10g and Oracle 11g on EMC® Symmetrix® DMX-3 and DMX-4 arrays using Virtual Provisioning.

June 2008

Part Number H4283.1

Table of Contents

Executive summary

EMC® Symmetrix® DMX-3 and DMX-4 storage arrays continue to provide increased storage utilization and optimization, enhanced capabilities, and greater interoperability and security, as well as multiple ease-of-use improvements. One such feature available with Enginuity™ release 5773 that provides increased storage utilization and optimization is Symmetrix Virtual Provisioning.

Virtual Provisioning, generally known in the industry as “thin provisioning,” enables organizations to improve ease of use, enhance performance, and increase capacity utilization for certain applications and workloads. The implementation of Virtual Provisioning for DMX-3 and DMX-4 storage arrays directly addresses improvements in storage infrastructure utilization, as well as associated operational requirements and efficiencies.

This white paper addresses the considerations for deploying Oracle 10g and Oracle 11g databases on thinly provisioned devices. An understanding of the principles that are exposed here will allow the reader to deploy Oracle 10g and Oracle 11g databases with Virtual Provisioning in the most effective manner.

Introduction

One of the biggest challenges facing storage administrators is storage provisioning for new applications. Administrators typically allocate space based on anticipated future growth of applications. This is done to mitigate recurring operational functions, such as incrementally increasing storage allocations or adding discrete blocks of storage as existing space is consumed. Using this approach results in more physical storage being allocated to the application than is needed for a significant amount of time and at a higher initial cost than is necessary. This overprovisioning of physical storage also leads to increased power, cooling, and floor space requirements. Even with the most careful planning, it may be necessary to provision additional storage in the future, which could potentially require an application outage.

A second layer of storage overprovisioning happens when a database administrator overallocates storage for a file or tablespace to ensure sufficient free space within the database. The operating system sees the space as completely allocated but internally only a fraction of the allocated space might be used.

EMC Virtual Provisioning can address both of these issues. Virtual Provisioning allows more storage to be presented to a host operating system than is physically available. More importantly, Virtual Provisioning will allocate physical storage only when the storage is actually written to. This allows more flexibility in predicting future growth and reduces the initial costs of provisioning storage to an application, and can obviate the inherent waste in overallocation of space and administrative management of subsequent storage allocations.

The Enginuity 5773 Release Notes will contain additional information about GA features and restrictions.

Audience

This white paper is intended for storage architects and administrators, server administrators, and database administrators responsible for deploying Oracle 10g and 11g on Symmetrix DMX-3 and DMX-4 using 5773 Enginuity with Virtual Provisioning.

Terminology

Table 1. Basic Symmetrix array terms

Term	**Description**
Device	A logical unit of storage defined within a Symmetrix array.
Device Capacity	The actual storage capacity of a device
Device Extent	The size of the smallest contiguous region of a device for which an extent mapping can occur

Host Accessible Device	A device that is exported for host use
Internal Device	A device used for internal function of the array
Metavolume	An aggregation of host accessible devices, seen from the host as a single device
Storage Pool	A collection of internal devices for some specific purpose

Table 2. Virtual Provisioning terms

Term	Description
Thin Device	A host accessible device that has no storage directly associated with it
Data Device	An internal device that provides storage capacity to be used by thin devices
Thin Device Extent	The minimum quantum of storage that must be mapped at a time to a thin device
Data Device Extent	The minimum quantum of storage that is allocated at a time when dedicating storage from a thin pool for use with a specific thin device
Extent Mapping	Specifies the relationship between the thin device and data device extents. The extent sizes between a thin device and a data device do not need to be the same
Thin Pool	A collection of data devices that provide storage capacity for thin devices
Thin Pool Capacity	The sum of the capacities of the member data devices
Bind	The process by which one or more thin devices are associated to a thin pool
Unbind	The process by which a thin device is diassociated from a given thin pool. When unbound, all previous extent allocations from the data devices are erased and returned for reuse
Enabled Data Device	A data device belonging to a thin pool on which extents can be allocated for thin devices bound to that thin pool.
Disabled Data Device	A data device belonging to a thin pool from which capacity cannot be allocated for thin devices.
Thin Pool Enabled Capacity	The sum of the capacities of enabled data devices belonging to a thin pool
Thin Pool Allocated Capacity	A subset of thin pool enabled capacity that has been allocated for the exclusive use of all thin devices bound to that thin pool
Thin Pool Preallocated Capacity	The initial amount of capacity that is allocated when a thin device is bound to a thin pool. This property is under user control
Thin Device Written Capacity	The capacity on a thin device that was written to by a host. In most implementations this is a subset of the thin device allocated capacity
Thin Device Subscribed Capacity	The total capacity that a thin device is entitled to withdraw from a thin pool. This may be equal to or less than the thin device capacity. In the current implementation they are equal
Thin Device Allocation Limit	The capacity limit that a thin device is entitled to withdraw from a thin pool, which may be equal to or less than the thin device subscribed capacity

Overview

Symmetrix thin devices are logical devices that can be used in many of the same ways that Symmetrix devices have traditionally been used. Unlike traditional Symmetrix devices, thin devices do not need to have physical storage completely allocated at the time the device is created and presented to a host. A thin device is not usable until it has been bound to a shared storage pool known as a thin pool. Multiple thin devices may be bound to any given thin pool. The thin pool is comprised of devices called data devices that provide the actual physical storage to support the thin device allocations.

When a write is performed to a part of any thin device for which physical storage has not yet been allocated, the Symmetrix allocates physical storage from the thin pool for that portion of the thin device only. The Symmetrix operating environment, Enginuity, satisfies the requirement by providing a block of storage from the thin pool called a thin device extent. This approach reduces the amount of storage that is actually consumed.

The minimum amount of physical storage that can be reserved at a time for the dedicated use of a thin device is referred to as a data device extent. An entire thin device extent is physically allocated to the thin device at the time the thin storage allocation is made as a result of a host write operation. The data device extent is allocated from any one of the data devices in the associated thin pool. Allocations across the data devices are balanced to ensure that an even distribution of allocations occurs from all available data devices in the thin pool.

The minimum amount of physical storage that must be mapped to a thin device at a time is called a thin device extent. For Symmetrix, the thin device extent size is the same as the data device extent size.

When a read is performed on a thin device, the data being read is retrieved from the appropriate data device in the thin pool to which the thin device is associated. If for some reason a read is performed against an unallocated portion of the thin device, zeroes are returned to the reading process.

When more physical data storage is required to service existing or future thin devices, for example, when a thin pool is approaching full storage allocations, data devices can be added to existing thin pools dynamically without needing a system outage. New thin devices can also be created and associated with existing thin pools.

When data devices are added to a thin pool they can be in an enabled or disabled state. In order for the data device to be used for thin extent allocation it needs to be in the enabled state. For it to be removed from the thin pool, it needs to be in a disabled state. A data device can be disabled only if it does not have any thin extents allocated.

The following figure depicts the relationships between thin devices and their associated thin pools. There are nine devices associated with thin Pool A and three thin devices associated with thin pool B.

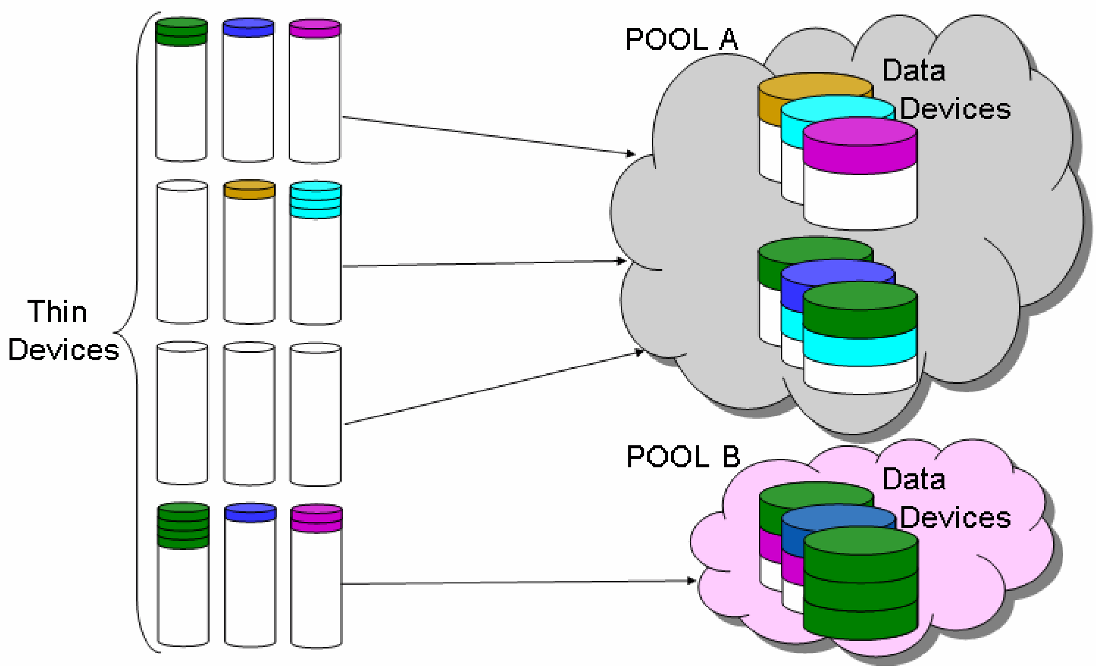

Figure 1. Thin devices and thin pools containing data devices

The way thin extents are allocated across the data devices results in a form of striping in the thin pool. The more data devices that are in the thin pool, then the wider the striping is, and the greater the number of devices that can participate in application I/O. The thin extent size for RAID 1 protected data devices is currently 768 KB in size. Future versions of the Enginuity operating environment may change the thin extent size.

The maximum size of a thin device is 64 GB. If a larger size is needed, then a metavolume comprised of thin devices can be created. It is recommended that the metavolume be concatenated rather than striped since the thin pool is already striped using data device extents. Concatenated metavolumes also support fast expansion capabilities, as new metavolume members can easily be appended to the existing concatenated metavolume. This functionality may be applicable when the provisioned thin device has become fully allocated at the host level, and it is required to further increase the thin device to gain additional space.

Striped metavolumes are supported with Virtual Provisioning and there may be workloads that will benefit from multiple levels of striping.

Requirements

Virtual Provisioning requires an Enginuity code level of 5773 or later. In order to create thin pools and thin devices and manage them, EMC Solutions Enabler 6.5 is required. If Symmetrix Management Console (SMC) is being used to manage Virtual Provisioning components, version 6.1 is needed. Thin pools can only be created by the customer and cannot be created during the bin file (DMX configuration) creation process.

Oracle and Virtual Provisioning

Oracle database files and Virtual Provisioning strategies

Oracle database file initialization

Using Virtual Provisioning in conjunction with Oracle databases provides the benefits mentioned earlier, such as reducing future server impact during LUN provisioning, increasing storage utilization, native striping in the thin pool, and ease and speed of creating and working with thin devices. However as commonly known, when Oracle initializes new files, such as log, data and temp files, it writes metadata to each initialized block. This will cause the thin pool to allocate the amount of space that is being initialized by the database. As database files are added, more space will be allocated in the pool. Due to Oracle file initialization, and in order to get the most benefit from a Virtual Provisioning infrastructure, a strategy for sizing files, pools, and devices should be developed in accordance to application and storage management needs. Some strategy options are explained next.

Oversubscription

An oversubscription strategy is based on using thin devices with a total capacity greater than the physical storage in the pool(s) they are bound to. This allows for optimizing storage utilization since the thin devices seem each to be a full-size device to the application while in fact only actual data is written to the shared thin pool. However, since Oracle database files initialize their space even though they are still empty, it is recommended that instead of creating very large data files that remain largely empty for most of their lifetime, smaller data files should be considered to accommodate near-term data growth. As they fill up over time, their size can be increased, or more data files added. In this case EMC recommends the use of the Oracle auto-extend feature for simplicity of management although DBAs may continue to use manual file management if they prefer.

An overprovisioning strategy is recommended for production environments when database growth is controlled, and thin pools can be actively monitored and their size increased when necessary in a timely manner.

Undersubscription

An undersubscription strategy is based on using thin devices with a total capacity smaller than the physical storage in the pool(s) they are bound to. This approach doesn’t necessarily increase storage utilization but still make uses of wide striping, thin pool sharing, and other benefits of Virtual Provisioning. In this case the data files can be sized to make immediate use of the full thin device size, or alternatively, auto-extend or manual file management can be used.

Undersubscribing is recommended when data growth is unpredictable, when multiple small databases share a large thin pool to benefit from wide striping, or when an overprovisioned environment is considered unacceptable.

Thin device preallocation

A third option exists where the DBA may like to benefit from overprovisioning and application sharing of the pool, but at the same time guarantee that space is reserved for some of the thin devices. This option uses thin device preallocation. A thin device can preallocate space in the pool, even before data was written to it. Figure 2 shows an example of creating 20 x 100 GB concatenated meta thin devices, and preallocating 20 GB in the pool for each of them. The example shows a Symmetrix Management Console screen; however a similar operation can be done using the Symmetrix CLI. When preallocation is used it is recommended to either preallocate the whole thin device (reducing the storage capacity optimization benefits) or preallocate a portion of the thin device, to match the size of the application file. For example, ASM disks can be set smaller than their actual full size, and later be resized dynamically without impacting the database application. In this case an ASM disk group can be created from these 20 thin metadevices, only using 20 GB of each disk. At a later time, additional storage on the thin device can be preallocated, and ASM disks resized to match it. Note that with the correct capacity planning and pool monitoring practices, thin device preallocation is not necessary in most production environments.

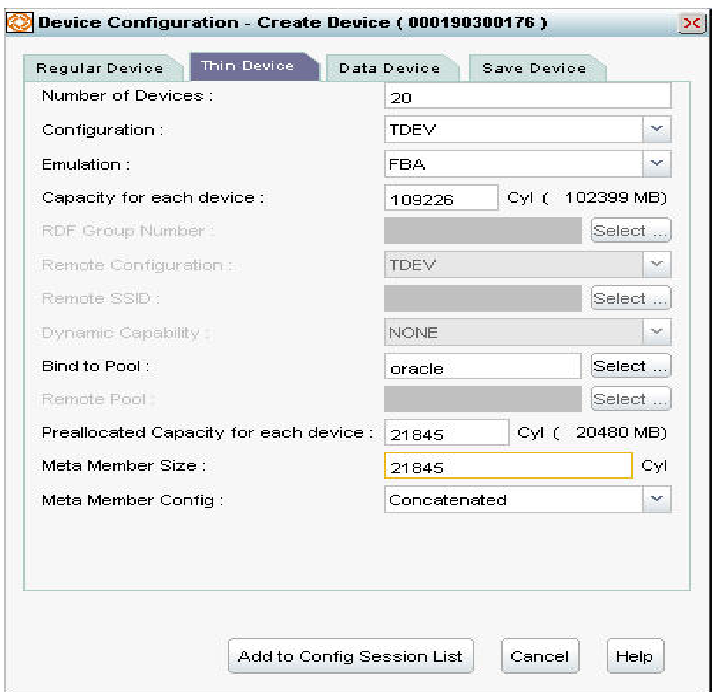

Figure 2. Symmetrix Management Console and thin device preallocation example

Thin device mapping and sizing summary

When planning the thin device sizes and quantity, the best approach is to map to the server as many thin devices as will possibly be needed for the lifetime of the database application on that server. Also their size should be as large as the long-term capacity growth calls for. These guidelines will help avoid future application impact due to additional LUN provisioning. In other words, it is common for the thin pool to be sized for *near-term* database capacity growth, and for the thin LUNs to be sized for *long-term* LUN capacity growth. Since the thin LUNs don't take space in the pool until data is written to them, this method optimizes storage utilization and reduces the database and application impact. Periodically, as database space requirements are reassessed and changes are made, these changes can be coordinated with the reassessment of the thin pool capacity and utilization.

Logical volume managers and file system considerations

It is important to remember that file systems and/or volume managers write metadata to the LUN, even though from the application perspective the volume/file system may look empty. The amount of metadata varies and while in some cases it is only a small signature, a header, or partition table, others can fill a significant portion of the LUN. Therefore not all logical volumes and/or file systems are equally optimized for use with thin pools. In this paper we'll explore Oracle Automatic Storage Management and OCFS2, which have both shown excellent compatibility with Virtual Provisioning.

Oracle Automatic Storage Management

Oracle Automatic Storage Management (ASM) provides capabilities for both a logical volume manager (striping, mirroring and disk management) and file system (directory structure and filenames). ASM is used by many customers to store the Oracle files as it natively stripes, restripes, and manages LUNs dynamically while the database is up and running.

When ASM was tested in conjunction with Virtual Provisioning it was found to be a good match. ASM metadata consumption was negligible in the thin pool as can be seen in Figure 3. In Figure 3 notice how the various file addition and deletion activities do not increase the overall amount of storage allocation in the pool. This indicates that ASM is reusing deleted space. This is important because unless thin devices are unbound, the thin pool doesn't release the allocated space (even if the data was deleted as far as the application is concerned). The differences in Figure 3 between ASM utilization and the thin pool are an attribute of oversubscription. The thin pool size reflects the actual available storage. The ASM disk group size reflects the total capacity of the thin devices, which is larger than the pool size in an oversubscription strategy. As data files were created, the ASM disk group reported more unused space than the thin pool, and therefore the difference in capacity utilization.

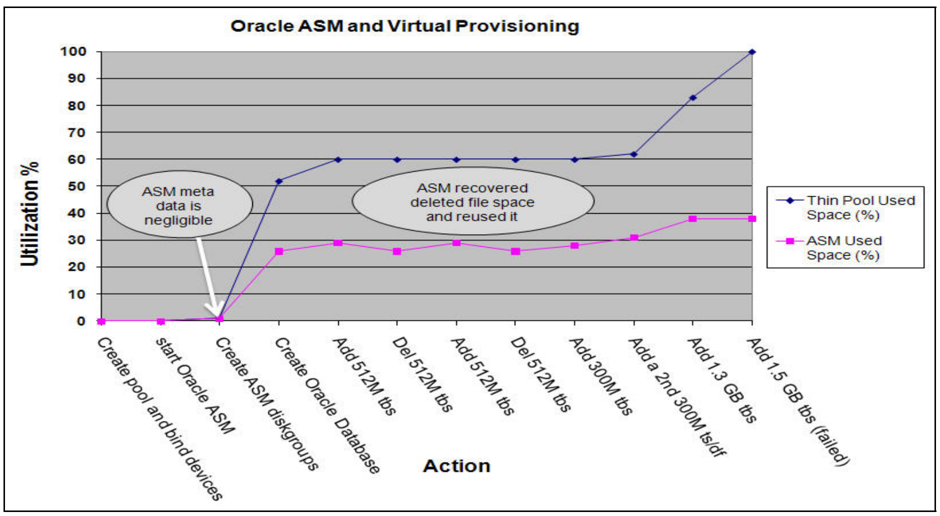

Figure 3. ASM and Virtual Provisioning

Oracle OCFS2

Oracle OCFS2 on Linux was also a good match with Virtual Provisioning. The results are shown in Figure 4. Also here, the differences in utilizations between the thin pool and OCFS2 are an attribute of the thin pool size relative to the size of the oversubscribed thin LUN, on which the OCFS2 file system was created.

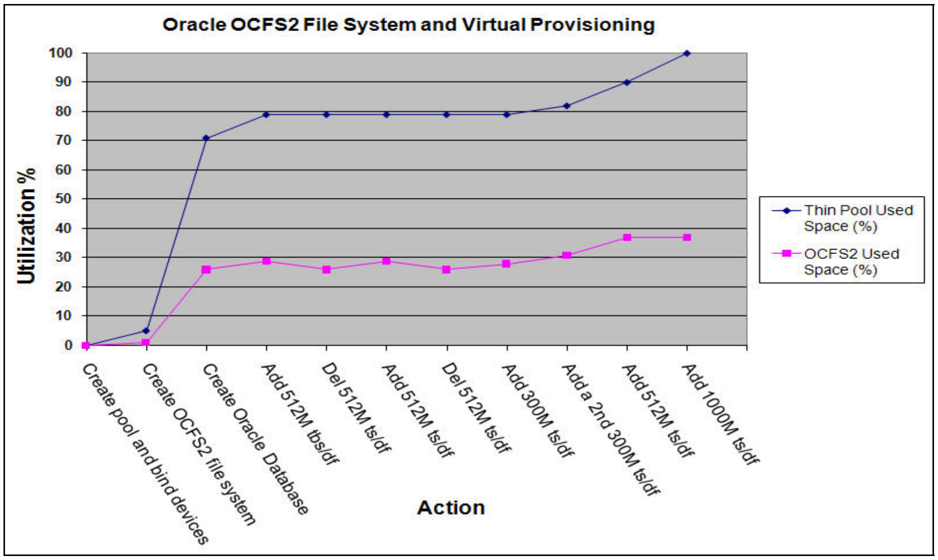

Figure 4. OCFS2 and Virtual Provisioning

Exhaustion of oversubscribed pool

An important aspect that Figure 3 and Figure 4 show is that when the thin pool was filled to 100% by adding a data file larger than the pool could accommodate (an event that should never happen in practice with correct planning and monitoring), the tablespace creation command failed. As a result both ASM and OCFS2 maintained their previously reported size and the database kept running transactions to previously allocated space.

Note that a situation where the pool has filled beyond its capacity should be avoided by correct monitoring and planning. However if a thin pool is oversubscribed and Oracle tries to write to a space in the pool that was not yet allocated, if a new extent can't be allocated in the pool, an I/O error will be returned to the server. In most cases the operation issuing the write, such as adding or extending a data file, will fail. However in some cases it may result in a file system hang and/or database crash. Since Oracle is an ACID-compliant database, customers should not be concerned that a crash, caused by the exhaustion of space in the thin pool, might cause data corruption. This will not happen with Oracle as there will be no loss of committed transactions. During database restart the database will undo and redo transactions to achieve a transactionally consistent point in time.

Layout and performance considerations

Each thin pool contains devices of similar RAID protection, size, and relative performance characteristics. A thin pool can also be shared by many thin devices with a total capacity smaller, equal, or larger than the actual physical space in the pool. While it is possible to create up to 512 thin pools and therefore separate applications to pools, it is recommended to use only a small number of pools. A single or small number of pools are easier to manage and monitor, and as a result each pool will have more devices, allowing better sharing of applications. Therefore it is recommended to add pools only when additional storage tiers are needed, or when certain applications can't share their storage[1].

From an Oracle layout planning perspective, it is common for OLTP databases to separate log file placement from data (to allow Symmetrix business continuity solutions like offloading backups, or fast database restores). Temp files are not required for disaster recovery and are likely to be placed on separate

[1] It is very common for Symmetrix logical devices and therefore applications to share the same physical disks. This is a concept that existed for many years before the introduction of Virtual Provisioning. So in fact, a shared pool is not a new idea or a radical change in a Symmetrix layout strategy.

devices. Also, Oracle Flash Recovery Area and Archive logs can be placed on a slower storage tier than the main database files. Therefore, outside the context of Virtual Provisioning, separation of log, data, temp, and archives/FRA was achieved by using different logical volumes (or metavolumes). In the context of Virtual Provisioning, this can be done by using thin devices in a similar way; however the thin pool can still be shared by the thin devices. The benefit of sharing the same pool is that Symmetrix business continuity and disaster recovery features work at the thin device level. This allows for the Oracle logs, for example, to benefit from the many devices (and therefore disk spindles) in the pool rather than be placed in another pool (or regular devices) with fewer disk spindles behind them. Only when a separate storage tier is required (such as for FRA and/or archive logs) should a different storage pool be used.

Figure 5 shows a small-scale OLTP workload. (It shows an Oracle 11g/ASM database of about 200 GB where both regular and thin devices were spread across the same number of RAID 1 volumes.) As can be seen, Virtual Provisioning showed a small performance benefit. In general, Virtual Provisioning provides a very good candidate for OLTP workloads by being natively striped across all the devices in the pool. In order to keep the application performance balanced, when increasing the pool size with additional devices, enough new devices should be added simultaneously so new data will be striped across them evenly.

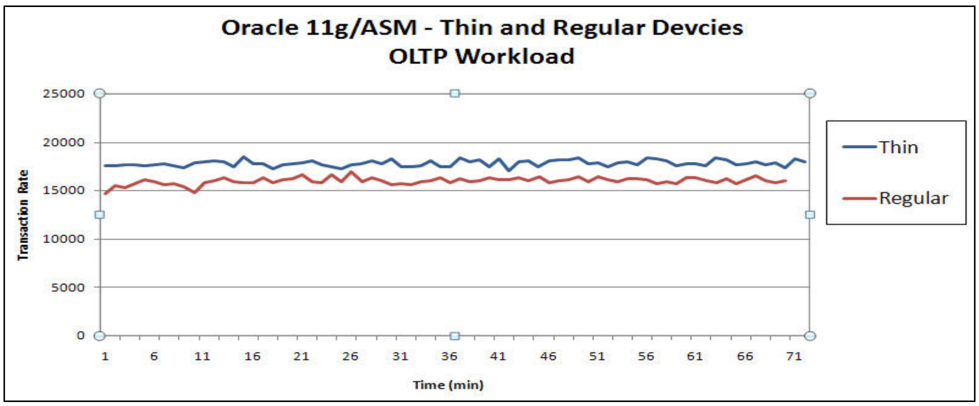

Figure 5. An OLTP workload running on thin devices

Thin pool monitoring

Along with Virtual Provisioning come several methodologies to monitor the capacity consumption of the thin pools. The Solutions Enabler 6.5 `symcfg monitor` command can be used to monitor pool utilization, as well as display the current allocations through the `symcfg show pool` option. There are also event thresholds that can be monitored through the SYMAPI event daemon, thresholds that can be set with Symmetrix Management Console, and SNMP traps that can be sent for monitoring by EMC ControlCenter® or any data center management product.

System administrators and storage administrators must put processes in effect to monitor the capacity for thin pools to make sure that they do not get filled. The pools can be dynamically expanded to include more data devices without application impact. These devices should be added in groups long before the thin pool approaches a full condition. If devices are added individually, hot spots on the disks can be created when much of the write activity is suddenly directed to the newly added data devices because other data devices in the pool are full.

Approaches for replication

Organizations will be able to perform "thin to thin" replication with Symmetrix thin devices by using standard TimeFinder, SRDF®, and Open Replicator operations. This includes TimeFinder/Snap, TimeFinder/Clone, SRDF/Synchronous, and SRDF/Asynchronous. Not all replication modes are available with the initial release. The Release Notes can provide more details.

In addition, thin devices can be used as control devices for hot and cold pull and cold push Open Replicator copy operations. If a push operation is done using a thin device as the source, zeroes will be sent for any regions of the thin device that have not been allocated, or that have been allocated but have not been written to.

Open Replicator can also be used to copy data from a standard device to a thin device. If a pull or push operation is initiated from a standard device that targets a thin device, then a portion of the target thin device, equal in size to the reported size of the source volume, will become allocated.

An example of working with Virtual Provisioning

Environment

- Symmetrix DMX-4 ID: 000190300176 (the scripts use "–sid 176" to refer to the Symmetrix)
- Host OS: Linux 64-bit kernel 2.6.18-8.el5
- Oracle 11g R1 using ASM
- EMC PowerPath® version 5.0.1

Setting up thin pool and devices

Create a thin pool called Oracle1 and allow it 500% oversubscription

```
# symconfigure -sid 176 -cmd "create pool Oracle1 type=thin max_subs_percent=500;" commit
```

Create 20 x 40 GB data devices and add them to the Oracle1 thin pool

```
# symconfigure -sid 176 -cmd "create dev count=20, size=40 GB, emulation=FBA, \
  in pool=Oracle1 member_state=ENABLE config=2-Way-Mir, attribute=datadev;" commit
```

Note: The output of the command includes the new device IDs:

```
...
New symdevs: 0072:0085
...
```

Create 20 x 59 GB thin devices and bind them to the Oracle1 thin pool

```
# symconfigure -sid 176 -cmd "create dev count=20, size=59 GB, emulation=FBA,
config=tdev binding to pool=Oracle1 ;" commit
```

Note: The output of the command includes the new device IDs.

```
...
New symdevs: 0132:0145
...
```

Host mapping/masking of the thin devices

Map the devices to four FA ports visible to the host

```
# symconfigure -sid 176 -cmd "map dev 0132:0145 to dir 1C:0 starting lun=32;" commit
# symconfigure -sid 176 -cmd "map dev 0132:0145 to dir 2C:0 starting lun=32;" commit
# symconfigure -sid 176 -cmd "map dev 0132:0145 to dir 15C:1 starting lun=32;" commit
# symconfigure -sid 176 -cmd "map dev 0132:0145 to dir 16C:1 starting lun=32;" commit
```

Mask the devices to the host if masking is used

```
# symmask -nop -sid 176 -wwn 10000000c9697b5d add devs 132:145 -dir 1C -p 0 -lun 7a
# symmask -nop -sid 176 -wwn 10000000c9697b5c add devs 132:145 -dir 2C -p 0
# symmask -nop -sid 176 -wwn 10000000c9697b53 add devs 132:145 -dir 15C -p 1
# symmask -nop -sid 176 -wwn 10000000c9697b52 add devs 132:145 -dir 16C -p 1
# symmask -nop -sid 176 refresh
```

Comment: Refresh the host devices by HBA online utilities or reboot.

List the new thin devices from the host

```
# symcfg disc
# powermt config
# sympd list
...
Symmetrix ID: 000190300176

        Device Name           Directors                  Device
--------------------------- ------------- -------------------------------------
                                                                          Cap
Physical                 Sym  SA :P DA :IT  Config        Attribute    Sts   (MB)
--------------------------- ------------- -------------------------------------
/dev/emcpowerj           0145 01C:0  NA:NA  TDEV          N/Grp'd      RW   60416
/dev/emcpowerk           0144 01C:0  NA:NA  TDEV          N/Grp'd      RW   60416
/dev/emcpowerl           0143 01C:0  NA:NA  TDEV          N/Grp'd      RW   60416
/dev/emcpowerm           0142 01C:0  NA:NA  TDEV          N/Grp'd      RW   60416
/dev/emcpowern           0141 01C:0  NA:NA  TDEV          N/Grp'd      RW   60416
/dev/emcpowero           0140 01C:0  NA:NA  TDEV          N/Grp'd      RW   60416
/dev/emcpowerp           013F 01C:0  NA:NA  TDEV          N/Grp'd      RW   60416
/dev/emcpowerq           013E 01C:0  NA:NA  TDEV          N/Grp'd      RW   60416
/dev/emcpowerr           013D 01C:0  NA:NA  TDEV          N/Grp'd      RW   60416
/dev/emcpowers           013C 01C:0  NA:NA  TDEV          N/Grp'd      RW   60416
/dev/emcpowert           013B 01C:0  NA:NA  TDEV          N/Grp'd      RW   60416
/dev/emcpoweru           013A 01C:0  NA:NA  TDEV          N/Grp'd      RW   60416
/dev/emcpowerv           0139 01C:0  NA:NA  TDEV          N/Grp'd      RW   60416
/dev/emcpowerw           0138 01C:0  NA:NA  TDEV          N/Grp'd      RW   60416
/dev/emcpowerx           0137 01C:0  NA:NA  TDEV          N/Grp'd      RW   60416
/dev/emcpowery           0136 01C:0  NA:NA  TDEV          N/Grp'd      RW   60416
/dev/emcpowerz           0135 01C:0  NA:NA  TDEV          N/Grp'd      RW   60416
/dev/emcpoweraa          0134 01C:0  NA:NA  TDEV          N/Grp'd      RW   60416
/dev/emcpowerab          0133 01C:0  NA:NA  TDEV          N/Grp'd      RW   60416
/dev/emcpowerac          0132 01C:0  NA:NA  TDEV          N/Grp'd      RW   60416
...
```

Align Linux partitions

Aligning partitions to a 64 KB offset is a general requirement on Linux x86_64 with ASM and Symmetrix storage arrays. Only the first partition requires alignment to 128 blocks (64 KB) to match the Symmetrix track size.

Use the command line to align a partition of a single device

Note: The text in **bold** is command input.

```
# fdisk /dev/emcpowerj
Command (m for help): n
p
Partition number (1-4): 1
First cylinder (1-60416, default 1): 1
Command (m for help): x
Expert command (m for help): b
Partition number (1-4): 1
```

```
New beginning of data (32-123731967, default 32): 128
Expert command (m for help): p

Disk /dev/emcpowerj: 64 heads, 32 sectors, 60416 cylinders

Nr AF  Hd Sec  Cyl  Hd Sec  Cyl     Start      Size ID
 1 00   1   1    0  63  32 1023       128  123731840 83
 2 00   0   0    0   0   0    0         0          0 00
 3 00   0   0    0   0   0    0         0          0 00
 4 00   0   0    0   0   0    0         0          0 00

Expert command (m for help): w
# chown oracle:dba /dev/emcpowerj1
```

Use a script to align the partitions of all thin devices

```
# cat ./asm_vp_part1.sh
#!/bin/bash
for i in j k l m n o p q r s t u v w x y z aa ab ac
do
  dd if=/dev/zero of=/dev/emcpower${i}1 bs=1024 count=1
  fdisk /dev/emcpower$i < ./fdisk.input
done

sleep 1

for i in j k l m n o p q r s t u v w x y z aa ab ac
do
  chown oracle.dba /dev/emcpower${i}1
done

# cat ./fdisk.input
d
n
p
1


x
b
1
128
p
w
```

Create an ASM diskgroup ready for an Oracle database

```
$ cat asmcre_vp1.sh
#!/bin/bash
export ORACLE_SID=+ASM
sqlplus  /NOLOG <<!
connect /as sysdba
startup nomount;

CREATE DISKGROUP REDO EXTERNAL REDUNDANCY DISK
 '/dev/emcpowerj1',
 '/dev/emcpowerk1',
 '/dev/emcpowerl1',
 '/dev/emcpowerm1';

CREATE DISKGROUP DATA EXTERNAL REDUNDANCY DISK
'/dev/emcpowern1',
'/dev/emcpowero1',
'/dev/emcpowerp1',
'/dev/emcpowerq1',
```

```
'/dev/emcpowerr1',
'/dev/emcpowers1',
'/dev/emcpowert1',
'/dev/emcpoweru1',
'/dev/emcpowerv1',
'/dev/emcpowerw1',
'/dev/emcpowerx1',
'/dev/emcpowery1',
'/dev/emcpowerz1',
'/dev/emcpoweraa1',
'/dev/emcpowerab1',
'/dev/emcpowerac1';
!

$ ./asmcre_vp1.sh

SQL*Plus: Release 11.1.0.6.0 - Production on Tue Mar 4 17:12:14 2008

Copyright (c) 1982, 2007, Oracle.  All rights reserved.

SQL> Connected to an idle instance.
SQL> ASM instance started

Total System Global Area  283930624 bytes
Fixed Size                  2143704 bytes
Variable Size             256621096 bytes
ASM Cache                  25165824 bytes
SQL> SQL>   2    3    4    5
Diskgroup created.

SQL> SQL>   2    3    4    5    6    7    8    9   10   11   12   13   14   15   16   17
Diskgroup created.

SQL> Disconnected from Oracle Database 11g Enterprise Edition Release 11.1.0.6.0 - 64bit
Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options
$
```

Recommendations

The following summarizes the optimal configurations of Virtual Provisioning and Oracle databases:

- Oracle databases using auto-extend
- Oracle databases that use manual file management and that increase the number and/or size of data files over time
- Systems where overallocation of storage is typical
- Systems where rapid growth is expected over time but downtime is limited

The following are situations where Virtual Provisioning may not be optimal:

- Systems where shared allocations from a common pool of thin devices is not desired
- Systems where data is deleted and re-created rapidly and the underlying file system or volume manager does not reuse the deleted space
- Systems that cannot tolerate an occasional response time increase of approximately 1 millisecond, due to writes to uninitialized blocks

Conclusion

EMC Symmetrix Virtual Provisioning provides a simple, noninvasive, and economical way to provide storage Oracle databases. Oracle ASM together with Oracle 10g or 11g was very synergetic to Virtual Provisioning due to its efficient metadata and behavior under an exhausted pool condition. The use of Oracle auto-extend or alternatively manual incremental growth of data files over time allows customers to derive the maximum benefits of Virtual Provisioning.

References

- *EMC Symmetrix Virtual Provisioning Implementation and Best Practices* technical note
- *New Features in EMC Enginuity 5773 for Symmetrix Open Systems Environments* white paper