EMC²
where information lives®

White Paper

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

Applied Technology

Abstract

This white paper provides a detailed review of the technical aspects and benefits of deploying Oracle database 10g and 11g on EMC® Symmetrix® VMAX™ arrays using Virtual Provisioning™.

August 2010

Part Number H4283.3

Table of Contents

Executive summary

The EMC® Symmetrix® VMAX™ is the newest addition to the Symmetrix product family. Built on the strategy of simple, intelligent, modular storage, it incorporates a new scalable Virtual Matrix™ interconnect that connects all shared resources across all VMAX Engines, allowing the storage array to grow seamlessly and cost-effectively from an entry-level configuration into the world's largest storage system. The Symmetrix VMAX provides improved performance and scalability for demanding enterprise storage environments while maintaining support for EMC's broad portfolio of platform software offerings

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 VMAX storage arrays directly addresses improvements in storage infrastructure utilization, as well as associated operational requirements and efficiencies.

Introduction

One of the biggest challenges facing storage administrators is storage provisioning for new applications. Incomplete performance and data growth rate requirements make storage layout and provisioning a challenging task. Administrators typically allocate space based on anticipated future growth of applications. This is done to mitigate recurring operational activities, 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 disruption.

Another case of unnecessary storage consumption happens when a database administrator preallocates large amounts of storage for data files or tablespaces 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.

Finally, designing a scalable storage layout based on isolating database objects to specific physical disks can be highly time-consuming and prone for ineffectiveness over the lifespan of the application due to changes. The desired approach is to spread the workload across all front-end and back-end resources allocated to the application rather than isolate the back-end resources for database objects. This approach allows even resource distribution, even as the application requirements change.

EMC Virtual Provisioning can address all of these issues. Virtual Provisioning allows more storage to be presented to a host operating system than is physically available, therefore removing the need to frequently provision the host with additional capacity. More importantly, Virtual Provisioning can allocate physical storage only when the storage is actually written to, allowing cost and energy savings. Virtual Provisioning also stripes the data across disks in the thin pool, regardless of whether a small or large thin device or database object is created, and therefore alleviates the need for spindle isolation and time-consuming storage layout planning. This type of storage layout is often referred to as "wide striping." When necessary, the thin pool can add storage devices and restripe the data seamlessly across them. Improved performance is gained with the Symmetrix prefetch algorithm that was updated to support thin device sequential reads.

Working with thin provisioning allows more flexibility in predicting future growth and reduces the initial costs of provisioning storage to an application. It can help obviate the inherent waste in preallocation of space and administrative management of storage layout planning and subsequent storage allocations.

Traditional data centers using thick storage LUNs can implement a thin provisioning strategy in phases by deploying EMC Virtual Provisioning for the copies of the production data and once the benefits are realized, deploy Virtual Provisioning for the production data. The thick-to-thin LUN replication introduced with the Enginuity™ 5874 Q4 2009 service release makes replication and transitioning of traditional thickly provisioned environments to Virtual Provisioning easy.

One of the goals when deploying Virtual Provisioning is to ensure the applications and migration tools used to store or move data from standard or fully provisioned environments to thin environments do not cause unneeded storage allocations. In some cases an application may write a contiguous series of zeros to represent available or initialized but unused space for a volume or file in a file system. When a standard environment is overallocated and a data file or volume has a high amount of unused space, an application that creates or copies this unused space to a thin device in the form of zeros can cause inefficient space utilization.

To help address this specific concern, Symmetrix Virtual Provisioning now enables Symmetrix VMAX users to automatically reclaim thin device extents containing all zeros. This is most beneficial after migrating from standard volumes to thin volumes, but can also prove beneficial in some application scenarios, in order to reduce capacity requirements and total cost of ownership (TCO). Note that Oracle RDBMSs do not initialize files with zeros and rather fill newly created empty files with block header information and metadata. Therefore Oracle database files will not directly benefit from zero space reclamation, but by following the specific guidelines described in this paper, storage that was once occupied by Oracle data files can be reclaimed before it is reused.

Oracle ASM disk groups on Symmetrix thin pool occupy only a small amount of storage for ASM meta data. In addition, ASM also reuses the storage consumed by prior Oracle objects. However, due to Oracle file initialization, users cannot reclaim allocated space, even on ASM controlled storage. Oracle has developed the ASM Storage Reclamation Utility (ASRU) that in conjunction with Symmetrix zero space reclamation allows thin pool storage reclamation for ASM-based Oracle objects. When some large Oracle ASM objects are deleted and space is no longer required, administrators can use ASRU and Symmetrix storage reclamation to reclaim space on the Symmetrix thin pool, thus realizing better storage utilization and cost savings.

EMC Virtual Provisioning allows for seamlessly adding more physical storage as the application demand grows. While increasing the available capacity, more physical disks can also improve application I/O performance because of wider striping and more back-end disk spindles to drive I/O traffic. The 5874 Q4 2009 service release of Symmetrix Enginuity introduced the automated pool balancing feature that allows restriping of existing application data to a new set of devices on the thin pool nondisruptively. While allowing added capacity, this aids in improving the overall application performance by balancing all I/O workloads across a greater number of physical disks.

This white paper provides a high-level overview of the Symmetrix VMAX architecture and some new features that allow improved storage provisioning, and easier management of storage allocation for Oracle databases on VMAX. The paper addresses several considerations for deploying Oracle database 10g and 11g on thinly provisioned devices. An understanding of the principles that are exposed here will allow the reader to deploy Oracle databases with Virtual Provisioning in the most effective manner. The paper also describes the Oracle ASM Storage Reclamation Utility (ASRU) and thin pool storage reclamation using Symmetrix zero space reclamation.

Audience

This white paper is intended for storage architects and administrators, server administrators, and database administrators responsible for deploying Oracle databases on Symmetrix VMAX with Virtual Provisioning.

Terminology

Table 1. Basic Symmetrix array terms

Term	Description
Device	A logical unit of storage defined within a Symmetrix array. In this paper this style of device is also referred to as a traditional device or a non-thin device
Device Capacity	The actual storage capacity of a device
Device Extent	Specifies a quantum of logical contiguous block of storage
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

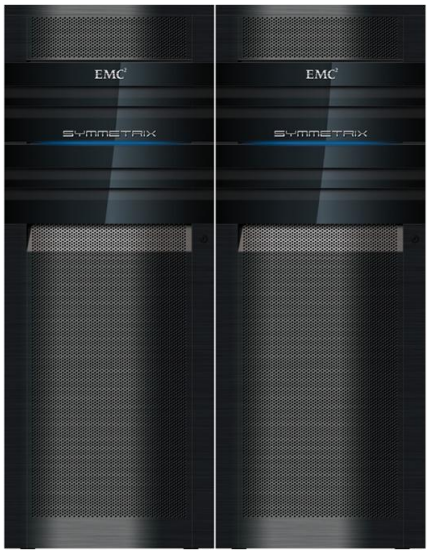
Term	Description
Thin Device	A host accessible device that has no storage directly associated with it
Data Device	An internal device that provides storage capacity to be used by thin devices
Thin Device Extent	The minimum quantum of storage that must be mapped at a time to a thin device
Data Device Extent	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. It should be noted that a thin pool can exist without any storage associated to it
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 allocated extents are freed and returned to the thin pool
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
Activated Data Device	Reads and writes can be done from allocated and unallocated space on activated data devices
Deactivated Data Device	Reads and writes can be done from already allocated space on deactivated data devices. No new allocations can be done on deactivated data devices

Data Device Drain	The process of removing allocated extents off of one data device and moving them to another enabled data device in the same thin pool
Thin Device Subscription Ratio	The ratio between the thin device subscribed capacity and the thin pool enabled capacity. This value is expressed as a percentage
Thin Device Allocation Ratio	The ratio between the thin device allocated capacity and the thin device subscribed capacity. This value is expressed as a percentage
Thin Device Utilization Ratio	The ratio between the thin device written capacity and thin device allocated capacity. This value is expressed as a percentage
Thin Pool Subscription Ratio	The ratio between the sum of the thin device subscribed capacity of all its bound thin devices and the associated thin pool enabled capacity. This value is expressed as a percentage
Thin Pool Allocation Ratio	The ratio between the thin pool allocated capacity and thin pool enabled capacity. This value is expressed as a percentage
Preallocating	Allocating a range of extents from the thin pool to the thin device at the time the thin device is bound. Sometimes used to reduce the operational impact of allocating extents or to eliminate the potential for a thin device to run out of available extents

Products and features overview

Symmetrix VMAX

Symmetrix VMAX, the newest member of the Symmetrix family, is a revolutionary new storage system purpose-built to meet all data center requirements as seen in Figure 1. Based on the Virtual Matrix Architecture™ and new Enginuity capabilities, Symmetrix VMAX scales performance and capacity to unprecedented levels, delivers nondisruptive operations, and greatly simplifies and automates the management and protection of information.



- 1 – 8 redundant V-Max Engines
- Up to 2.1 PB usable capacity
- Up to 128 FC FE ports
- Up to 64 FICON FE ports
- Up to 64 Gig-E / iSCSI FE ports
- Up to 1 TB global memory (512 GB usable)
- 48 – 2,400 drives
- Enterprise Flash Drives 200/400 GB
- FC drives 146/300/450 GB 15k rpm
- FC drives 300/450/600 GB 10k rpm
- SATA drives 1 TB 7.2k rpm

Figure 1. The Symmetrix VMAX platform

The Symmetrix VMAX design is based on individual engines with redundant CPU, memory, and connectivity on two directors for fault tolerance. VMAX Engines connect to and scale out through the Virtual Matrix Architecture, which allows resources to be shared within and across VMAX Engines. To meet growth requirements, additional VMAX Engines can be added nondisruptively for efficient and dynamic scaling of capacity and performance that are available to any application on demand.

Symmetrix Auto-provisioning Groups

Oracle Real Application Clusters (RAC) requires shared storage across all RAC nodes and as new nodes are added or removed storage should remain visible to all new sets of RAC nodes. Oracle Grid builds upon the multiple applications, storage and server tiers and new nodes are added dynamically to support changing demands of the Grid infrastructure. Virtual infrastructures using Oracle VM and VMware allow dynamic movement of virtual machines across host boundaries to provide load balancing and fault tolerance for the applications. For all such Oracle use cases, the ability to dynamically make storage visible to new nodes and grow the available capacity on demand is a critical requirement. Traditional storage provisioning mechanisms rely on storage masking and mapping to individual hosts and it would not be easy to keep up with changing environments.

Enginuity version 5874 and later provide storage and system administrators with a simplified model for storage provisioning referred to as Auto-provisioning Groups. Auto-provisioning Groups are implemented using the `symaccess` CLI command with Solutions Enabler 7.0 and later or by using the Symmetrix Management Console (SMC).

The fundamental concept of Auto-provisioning Groups is the logical grouping of related objects and the creation of a view that associates the related groups together. The following logical groupings are used:

- **Initiator groups.** An initiator group is a logical grouping of up to 32 Fibre Channel initiators (HBA ports) identified by World Wide Names or WWNs or eight iSCSI names or a combination of both. An initiator group may also contain the name of another initiator group to allow the groups to be cascaded to a depth of one.
- **Port groups.** A port group is a logical grouping of Fibre Channel and/or iSCSI Symmetrix front-end director ports.
- **Storage groups**. A storage group is a logical grouping of up to 4,096 Symmetrix devices. Both thick and thin LUNs can be used.
- **Masking views.** A masking view defines an association between one initiator group, one port group, and one storage group. When a masking view is created, the devices in the storage group are mapped to the ports in the port group and masked to the initiators in the initiator group. Depending on the server and application requirements, each server or group of servers may have one or more masking views that associate a set of Symmetrix devices to an application, server, or cluster of servers.

With Auto-provisioning Groups, related initiators (HBAs) are grouped into an initiator group, related front-end ports are grouped into a port group, and related devices are grouped into a storage group. A masking view associates the groups together. At the time the masking view is created, all the required mapping and masking are performed automatically in a single operation. This approach dramatically reduces complexity and simplifies storage allocation.

Oracle databases can highly benefit from Auto-provisioning Groups as the feature makes storage device provisioning, monitoring, and management very easy. For example, for Oracle Real Application Clusters storage devices are simply added to the storage group and automatically masked to all the servers (initiator groups) in the cluster. In the Oracle Grid model as servers and applications require changes in access to storage devices, host HBA ports, or storage connectivity, the use of Auto-provisioning Groups makes this an easy and fast task to manage.

Figure 2 shows an example of using Auto-provisioning Groups to mask Oracle RAC database devices. A storage group is created with the database devices and a port group with the Symmetrix ports. A single initiator group can be created for all hosts' HBAs (or each host can have its own initiator group and they will be cascaded for ease of management and scalability). The Auto-provisioning Groups masking view simply includes the storage group, port group, and initiator group. If any hosts are added or removed from the cluster they will simply be added or removed from the initiator group. In a similar way, devices or Symmetrix ports can be added or removed from their groups and the view will automate the device provisioning for the cluster.

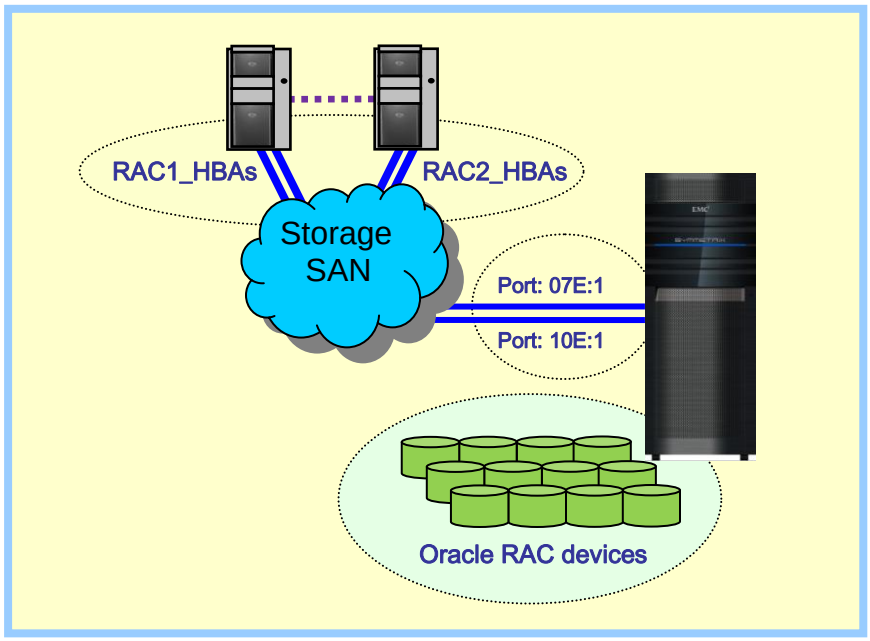


Figure 2. Oracle RAC and Auto-provisioning Groups

The following steps demonstrate the use of Auto-provisioning Groups, based on the example in Figure 2.

1. **Create** a **storage group for RAC devices**
   symaccess -name RAC_devs -type storage devs 790:7AF create
2. **Create a port group with storage ports** 7E:1 and 10E:1
   symaccess -name RAC_ports -type port –dirport 7E:1,10E:1 create
3. **Create an initiator group for each cluster node's HBAs**
   symaccess -name RAC1_hbas -type initiator -file ./RAC1_hbas.txt create
   The file RAC1_hbas.txt contains: WWN:10000000c975c2e4
   WWN:10000000c975c336
   symaccess -name RAC2_hbas -type initiator -file ./RAC2_hbas.txt create
   The file RAC2_hbas.txt contains: WWN:10000000c975c31a
   WWN:10000000c975c3ab
4. **Cascade the cluster nodes' initiator groups into a single one for the entire cluster**
   symaccess -name RAC_hbas -type initiator create
   symaccess -name RAC_hbas -type initiator add –ig RAC1_hbas
   symaccess -name RAC_hbas -type initiator add –ig RAC2_hbas
5. **Create the view for the entire RAC cluster storage provisioning**
   symaccess create view -name RAC_view
   -storgrp RAC_devs -portgrp RAC_ports -initgrp RAC_hbas

Symmetrix Management Console

Many large enterprise data centers have stringent change control processes that ensure reliable execution of any modification to their IT infrastructure. Often changes are implemented using scripts that are fully documented, have been thoroughly reviewed, and can be consistently executed by all storage administrators. An alternative to using scripts is to use the Symmetrix Management Console (SMC).

The SMC, as shown in Figure 3, is a graphical user interface (GUI) that allows storage administrators to easily manage a Symmetrix. SMC can be run on supported open systems hosts connected directly or remotely to a Symmetrix VMAX that requires management.

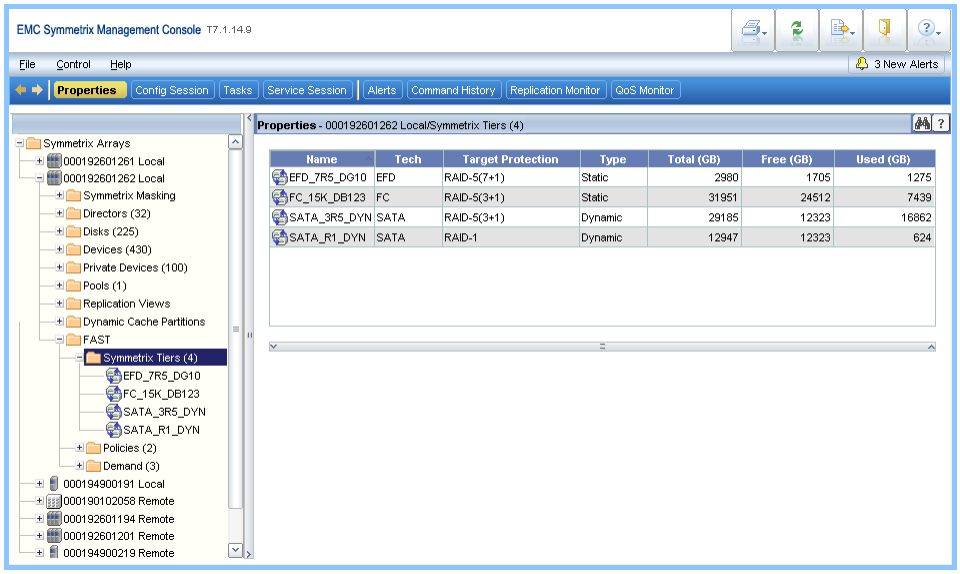


Figure 3. Symmetrix Management Console

Overview of Symmetrix VMAX Virtual Provisioning

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 preallocated at the time the device is created and presented to a host (although in some cases customers interested only in the thin pool wide striping and ease of management choose to fully preallocate the thin devices). 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. 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.

For Symmetrix, the thin device extent size is the same as the data device extent size, which is 12 Symmetrix tracks or 768 KB in size. There is no waste associated with this approach as normally applications allocate large portions of data at a time, and even if the data is striped across multiple LUNs, it will be stored in an adjacent location every time the striping mechanism will wrap over the same LUN, essentially filling up the thin pool extents. For example, Oracle data files are usually created with a size range of a single gigabyte to tens of gigabytes. Even when a host logical volume manager (LVM) is used to stripe the data, such as Oracle ASM, each LUN that is used (or ASM member) will create the next Allocation Unit adjacent to the previous one. In accordance, the thin devices will fill up data device extents completely before moving to the next one in the thin pool. That way the data is striped in the thin pool evenly and without waste.

As a note, there is no reason to match the LVM stripe depth with the thin device extent size. Oracle accesses data either by random single block read/write operations (usually 8 KB in size) or sequentially by reading large portions of data. In either case there is no advantage or disadvantage to match the LVM stripe depth to the thin device extent size as single block read/write operates on a data portion that is smaller than the LVM stripe depth anyway, and sequential operations will simply continue to read data on each LUN (every time the sequential read wraps to that same LUN) regardless of the stripe depth.

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, zeros 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. Symmetrix automatically initiates a drain operation on a disabled data device without any disruption to the application. Once all the allocated extents are drained to other data devices, a data device can be removed from the thin pool.

The following figure depicts the relationships between thin devices and their associated thin pools. The thin Pool A contains nine data devices, and thin Pool B contains three data devices. There are nine thin devices associated with thin Pool A and three thin devices associated with thin pool B. The data extents for thin devices are distributed on various data devices as shown in Figure 4.

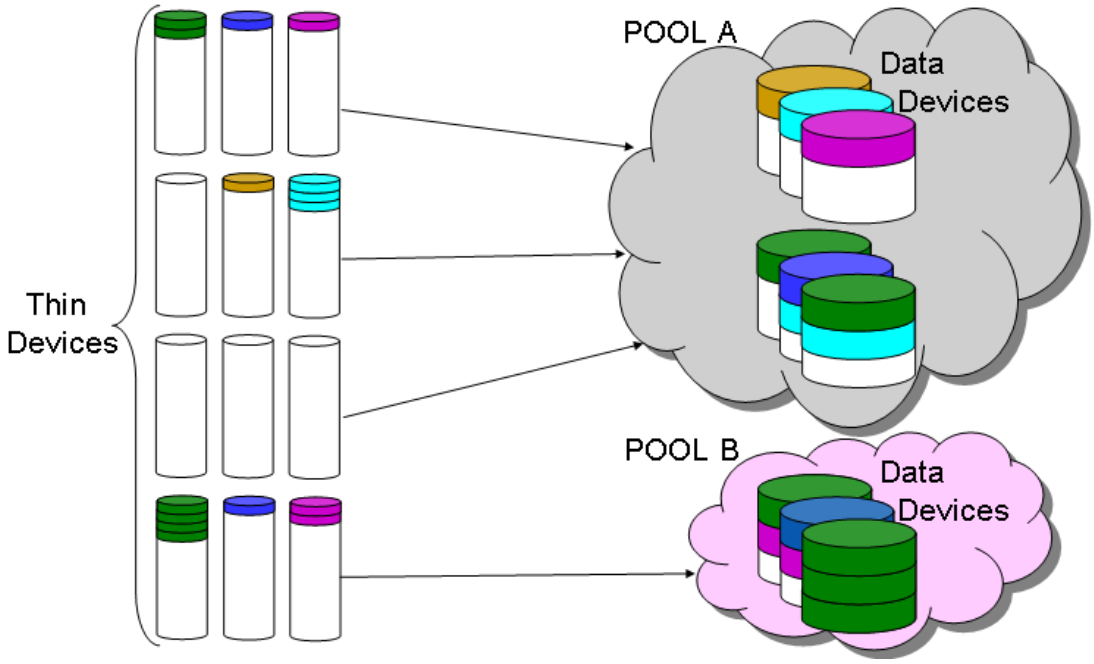


Figure 4. Thin devices and thin pools containing data devices

The way thin extents are allocated across the data devices results in a form of striping in the thin pool. The more data devices in the thin pool, the wider the striping will be, and the greater the number of devices that can participate in application I/O. As mentioned earlier, the thin extent size for data devices is currently 12 tracks or 768 KB in size. Future versions of the Enginuity operating environment may change the thin extent size.

The maximum size of a thin device in a Symmetrix VMAX is 240 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 be easily 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. Note that it is not recommended to provision applications with a low number of very large LUNs. The reason is that each LUN provides the host with an addition I/O queue to which the host operating system can stream I/O requests and parallelize the workload. Host software and HBA drivers tend to limit the amount of I/Os that can be queued at a time per LUN and therefore to avoid host queuing bottlenecks under heavy workloads, it is better to provide the application with multiple, smaller LUNs rather than very few and large LUNs.

Striped metavolumes are supported with Virtual Provisioning and there may be workloads that will benefit from multiple levels of striping (for example, for Oracle redo logs when SRDF®/S is used, and host striping is not available).

Data devices in the thin pool can be protected with any EMC RAID offering: RAID 1, RAID 5, or RAID 6. Both RAID 1 and RAID 5 protect from a single-drive failure, and RAID 6 protects from two-drive failures. A RAID 1 group resides on two physical disk drives; a RAID 5 (3+1) group resides on four disk physical drives, and so on. When a thin pool of disk is created it is always created out of similarly configured RAID groups and not individual disks. For example, if we create eight RAID 5 (3+1) thin devices and put them into one pool, the pool has eight RAID 5 devices of four devices each. If one of the disks in this pool fails, you are not losing one disk from a pool of 32 devices, rather you are losing one disk from one of the eight RAID devices and that RAID group can continue to service read and write requests, in degraded mode, without data loss. Also, as with any RAID group, with a failed disk Enginuity will immediately invoke a hot sparing operation to restore the RAID group to its normal state. While this RAID group is rebuilding, any of the other RAID groups in the pool can have a disk failure and there is still no loss of data. In this example, with eight RAID groups in the pool there can be one failed disk in each RAID group in the pool without data loss. In this manner data stored in the thin pool is no more vulnerable to data loss than any other data stored on similarly configured RAID devices. Therefore a protection of RAID 1 or RAID 5 for thin pools is acceptable for most applications and RAID 6 is only required when in situations where additional parity protection is warranted.

Symmetrix VMAX Virtual Provisioning automated pool rebalancing

Starting with the Enginuity 5874 Q4 2009 service release and Solutions Enabler 7.1, automated pool rebalancing allows the user to run a balancing operation that will redistribute data evenly across the enabled data devices in the thin pool. Because the thin extents are allocated from the thin pool in round-robin fashion, the rebalancing mechanism will be used primarily when adding data devices to increase thin pool capacity. If the automated pool rebalancing is not used, existing data extents will not benefit from the added data devices as they will not be redistributed.

The balancing algorithm will calculate the minimum, maximum, and mean used capacity values of the data devices in the thin pool. The Symmetrix will then move thin device extents from the data devices with the highest used capacity to those with the lowest until the pool is balanced. Symmetrix VMAX automated pool rebalancing allows nondisruptive extension of a thin pool in increments as needed, maximizing the performance and minimizing TCO.

Virtual Provisioning natively offers wide striping and balanced disk access across the enabled data devices in the pool. Automated pool rebalancing allows redistribution of data extents in the thin pool when new devices are made available. The balanced distribution of data extents over a larger set of data devices would result in balanced disk utilization at the Symmetrix back end. This helps achieve higher overall application performance as described in the following use case.

Example of achieving higher Oracle database performance with automated pool rebalancing

Methodology and configuration

The following is an example of testing an Oracle database using Symmetrix VMAX Virtual Provisioning with the automated pool rebalancing feature. The test included multiple phases and each phase involved increasing the storage capacity in the thin pool by adding data devices and rebalancing the pool. The identical database workload was run during each phase and database I/O performance data was captured. The Oracle database transaction rates were used to analyze the effect of automated pool rebalancing on Oracle database I/O performance.

In this test case a 1 TB Oracle database was created on one thin pool comprising 35 thin LUNs as shown in Table 3. The tests were performed while running an OLTP workload (approximately 30 percent writes and a 70 percent random read mix). The thin pool data devices were protected using a RAID 5 (3+1) configuration (four physical disks used per data device). The number of data devices in the thin pool was gradually changed from four to 12 with an increment of four data devices in every phase and effectively increased the number of spindles by 16 (a new set of four physical disks used for every data device) in every phase to drive the database workload. The number of thin LUNs provisioned to the Oracle database remained the same (35) throughout the tests and no changes were made to the database or host environment. Symmetrix VMAX Virtual Provisioning allowed nondisruptive addition of new data devices in the thin pool. Automated pool rebalancing was used after adding a new set of data devices to the pool and the next test performed after the rebalancing operation completed.

Table 3. Initial test configuration

Configuration	Description
Storage array	Symmetrix VMAX SE (single engine)
Enginuity	5874 Q4 2009 service release
Oracle	CRS and database version: 11g R2
Logical Volume Manager	Oracle ASM
Thin LUNs	35 x 30 GB
Data Devices	4 x 240 GB - RAID 5 (3+1)
Thin Pool	Oracle1
Linux	Red Hat Linux 5.2

Execution

The test case was executed using the three phases described below. The Oracle database used separate ASM disk groups for DATA, REDO, ARCH, and TEMP and all used the ASM disk members from the thin pool Oracle1. For more complete details of the Solutions Enabler command line used during the test execution please refer to Appendix A for an example of working with Virtual Provisioning.

In phase I of the test, four data devices serviced the thin pool. A transactions per minute (TPM) rate of around 3,000 was reported for the OLTP workload.

In phase II, four more data devices were added to the thin pool, doubling the total number of physical spindles supporting the workload to 32. After adding new data devices, automated pool rebalancing was performed. The data extents were balanced across all 32 data devices. Once the pool balancing operation completed, the OLTP workload was run and a TPM rate of around 4,900 was reported for the workload.

In phase III, four more data devices were added to the thin pool, meaning the total number of physical spindles supporting the workload was 48. After adding new data devices, automated pool rebalancing was performed. The data extents were balanced across all 48 data devices. Once the pool balancing operation completed, the OLTP workload was run and a TPM rate of around 5,700 was reported for the workload, indicating that disks are no longer a bottleneck for this workload.

Monitoring

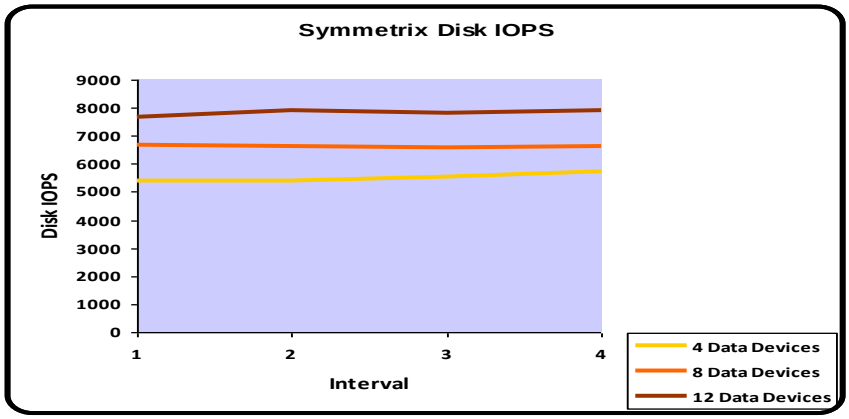
Symmetrix statistics were collected using STP (or Workload Analyzer), Oracle statistics were collected using AWR reports, and TPM data was captured from the database benchmark tool. Either SYMCLI or SMC can be used to manage the thin pool, adding and removing data devices, and to monitor progress information related to pool balancing.

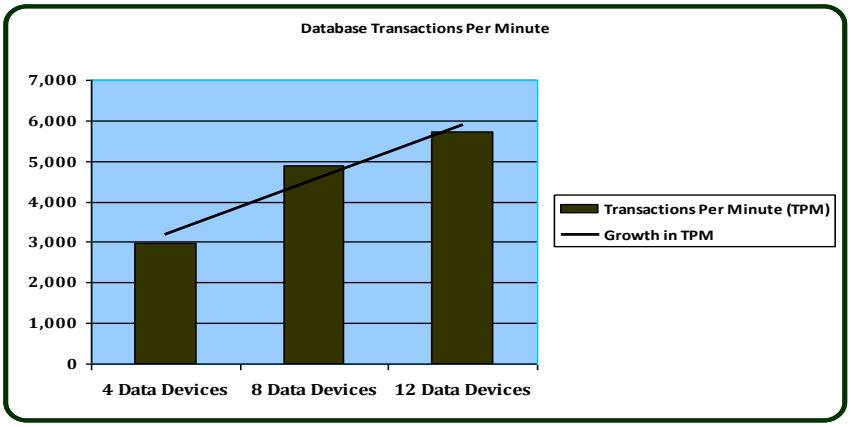
Results

In this example, the same number of thin devices (35) was used during all the tests; the only change was in the data devices serving the thin pool. Thus without any downtime for the application, capacity was added to the pool, and additional physical disks and thus extra spindles were available to drive the database I/O, effectively lowering the disk response time and improving the database transaction rate.

Oracle database performance was achieved as depicted in Figure 5, and it should be noted here that the entire application I/O stack is affected by multiple components, including host CPU, memory, HBAs, I/O queuing, the SAN, and the storage array. Therefore, relieving a storage I/O bottleneck does not always result in an improved application transaction rate. However, using automated pool rebalancing makes it easy to manage storage bottlenecks that otherwise often require application layout changes.

This use case shows that automated pool rebalancing can improve database I/O performance while nondisruptively making more physical storage capacity available for thin devices.





	4 Data Devices	8 Data Devices	12 Data Devices
Database Transactions Per Minute (TPM)	2,987	4,906	5,716
% Improvement in TPM over 4 Data Devices		64.24%	91.34%

Figure 5. Oracle transactions per minute (TPM) improvement with the number of data devices in the thin pool

Oracle and Symmetrix 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 capacity utilization since the thin devices each seem to be a full-size device to the application, while in fact the thin pool can't accommodate the total LUNs' capacity. 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 conjunction with the capacity increase of the thin pool. The Oracle auto-extend feature can be used for simplicity of management, or DBAs may prefer to use manual file size management (or addition).

An oversubscription strategy is recommended for database 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 improve storage capacity utilization but still makes use 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 oversubscriptioned environment is considered unacceptable.

Thin device preallocation

A third option exists where the DBA may like to benefit from oversubscription and application sharing of the pool, but at the same time guarantee that space is reserved for some (or all) 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 6 shows an example of creating 10 x 29.30 GB (32,000 cylinders per device) thin devices, and preallocating 10 GB in the pool for each of them. The example shows a Symmetrix Management Console screen (a similar operation can be done using the Symmetrix CLI). When preallocation is used customers often preallocate the whole thin device (reducing the storage capacity optimization benefits). In effect each thin device therefore fully claims its space in the thin pool, eliminating possible thin pool out-of-space condition. It is also possible to preallocate a portion of the thin device (like the 10 GB in the example) 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 any disruption to the database application. In this case an ASM disk group can be created from these 10 thin devices, only using 10 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.

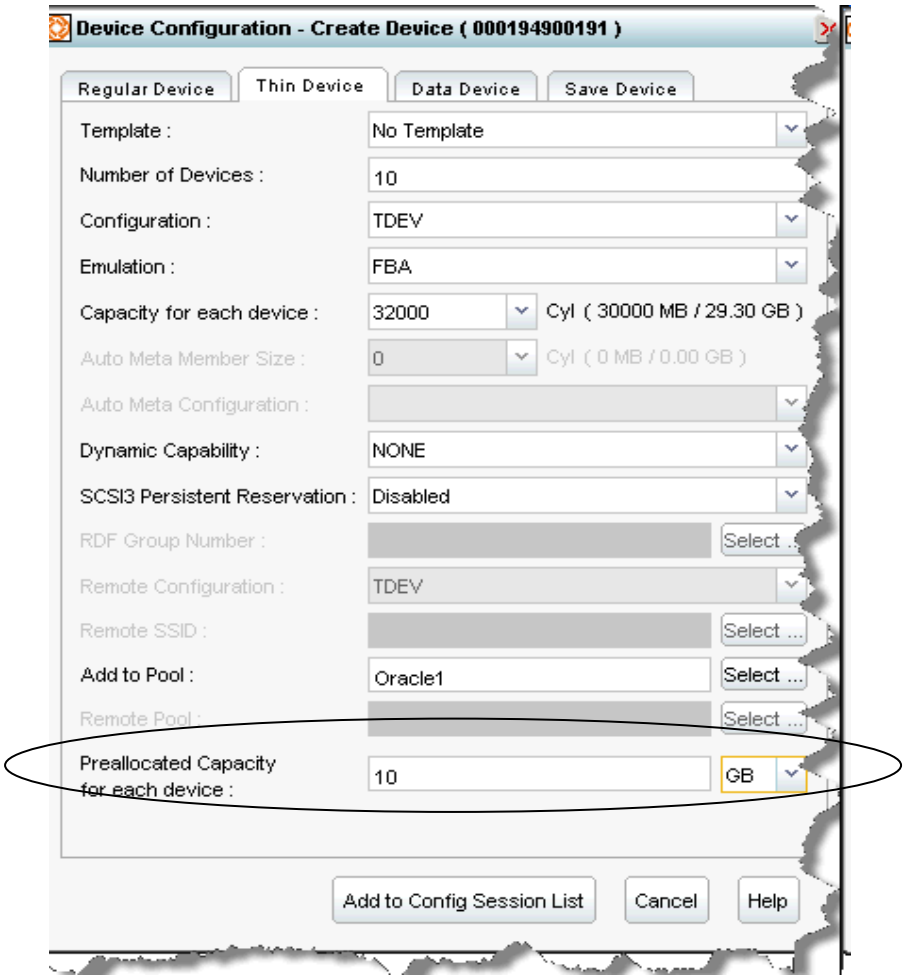


Figure 6. Symmetrix Management Console and thin device preallocation example

Thin device mapping and sizing summary

When planning the thin device sizes and quantities, the best approach when using oversubscription is to map to the server as many thin devices as will 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 capacity utilization and reduces the database and application impact as they continue to grow. Periodically, as database space and performance requirements are reassessed, and changes are made, these changes can be coordinated with the reassessment and changes to the thin pool.

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, or 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 (ASM) and OCFS2, which have both shown excellent compatibility with Virtual Provisioning.

Oracle Automatic Storage Management

Oracle 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.

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 7. In the figure 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 7 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.

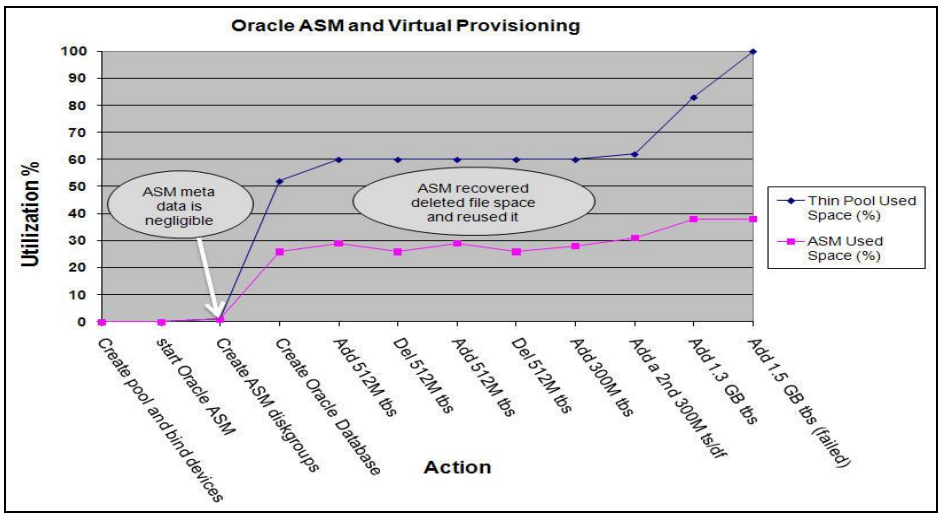


Figure 7. ASM and Virtual Provisioning

Oracle OCFS2

Oracle OCFS2 on Linux was also a good match with Virtual Provisioning. The results are shown in Figure 8. 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.

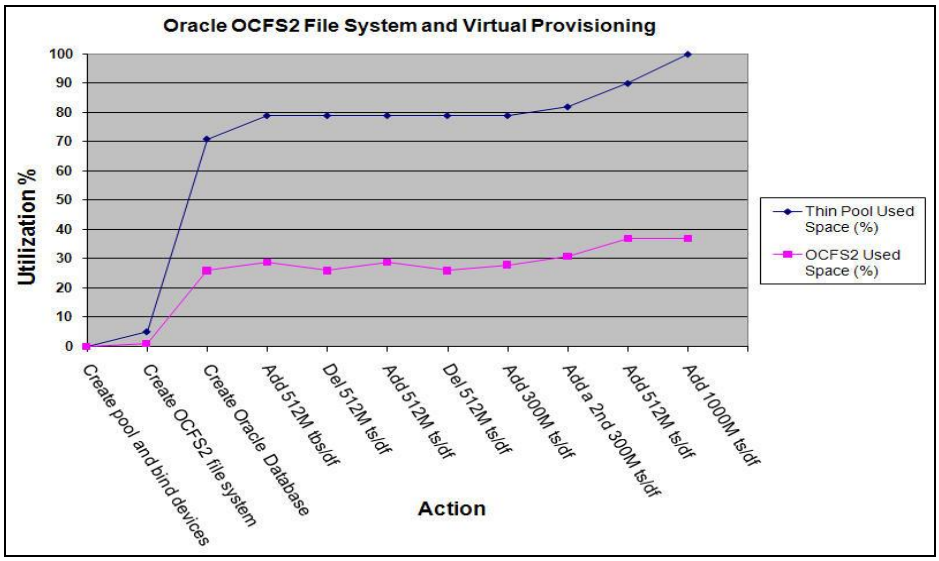


Figure 8. OCFS2 and Virtual Provisioning

Exhaustion of oversubscribed pool

An important aspect that Figure 7 and Figure 8 show is that when the thin pool was filled to 100 percent 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 host. 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, after appropriate storage is added to the exhausted thin pool, a database restart will automatically recover all committed transactions up to the point of failure.

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 storage resources. 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 devices. Also, the Oracle Flash/Fast[2] Recovery Area and Archive logs can be placed on a slower storage

[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.
[2] Oracle renamed Flash Recovery Area to Fast Recovery Area with database release 11g R2.

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 that require similar RAID protection and performance requirements. 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. When a separate storage tier is required (such as for FRA and/or archive logs) a different thin pool should be used.

Figure 9 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 physical disks. 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. Starting with the Enginuity 5874 Q4 2009 service release, the number of devices added simultaneously is less important, as long as a pool rebalance takes place afterward.

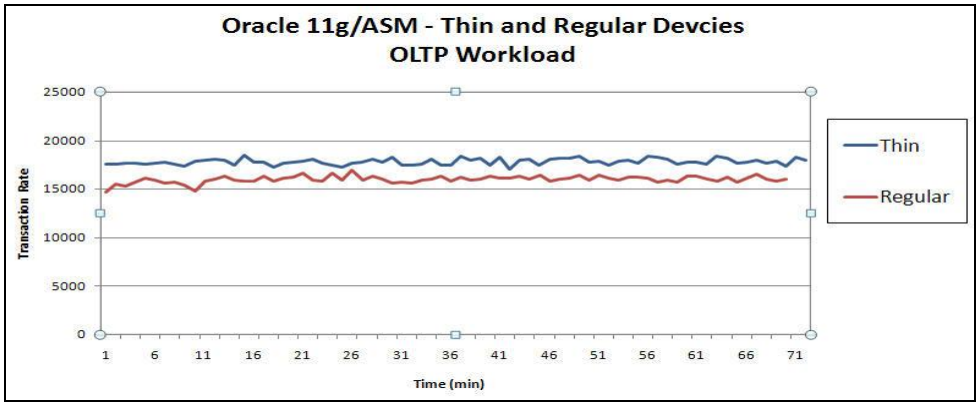


Figure 9. An OLTP workload running on thin devices

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 Ionix™ ControlCenter® or any data center management product.

If oversubscription is used in conjunction with Virtual Provisioning then 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, unless the thin pool is rebalanced 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.

ASM rebalance and Virtual Provisioning rebalance strategies

Oracle ASM performs rebalancing at the host level by moving ASM extents across a new set of devices whenever the number of available ASM devices (members) changes. An ASM rebalancing operation considers device capacity and uses the initialization parameter of rebalance power to determine how many processes participate when performing such an operation. The operation is performed nondisruptively. An ASM rebalance is based on capacity considerations alone and it does not take in account the frequency of access to data or I/O performance. Hence ASM rebalancing helps with spreading data evenly across available devices in a single ASM disk group.

EMC Virtual Provisioning automated pool rebalancing on the other hand works at the Symmetrix level. By adding data devices to the thin pool and balancing the pool, data device extents can be spread across a larger number of physical disk spindles, which drives higher disk performance without having to change anything by the host or ASM. If additional capacity is required by the application, new thin devices can be created and added to the application.

Thus ASM rebalance and EMC Virtual Provisioning pool balancing are complementary – ASM allows host-level load balancing and EMC Virtual Provisioning pool balancing allows storage-level load balancing for database I/O.

Reclamation of space and the ASM Storage Reclamation Utility (ASRU)

One of the goals for deploying Virtual Provisioning is to improve storage capacity utilization to make unused host LUNs' storage space available for other uses. Applications, however, have different characteristics with regard to storage allocation and release. Some applications reserve the storage by initializing the space for guaranteed availability, and others defer the allocation. Most applications, though, don't zero out the space when the allocated storage is no longer needed (for example, files are deleted) and in some cases, for the chance the user would like to undelete a file, the application would allocate new space on the LUN rather than reuse the deleted files' space (Windows NTFS is such an example). As time progresses there is a potential of having a lot of allocated but un-needed space on the thin pool. It becomes necessary to reclaim such space and make it available for other uses. The Enginuity 5874 Q4 2009 service release introduced zero space reclamation that allows reclaiming data extents containing contiguous zero blocks. The feature currently works at the granularity of 12 tracks (768 KB), which may change in the future. It becomes necessary to zero out the space containing stale application data to make extents candidates for reclamation. By following some best practices when an application no longer needs the storage, very efficient capacity utilization and hence space and energy savings can be realized in data centers.

Reclamation of space used by Oracle objects

Existing Oracle objects on the thin pool

As described earlier in the paper, Oracle database file initialization results in full allocation of the thin pool extents that the database files occupy. That is because Oracle blocks headers and metadata information that was written at the time of the file initialization. Therefore, empty data files will not have long sequences of zeros that can be reclaimed in the thin pool (and redo logs are cyclical and therefore no reason to reclaim their space). For that reason if using oversubscription approach, it is best to create Oracle data files sized for a short- to mid-term capacity growth plan and expand or add files over time.

Dropped or deleted Oracle objects

When database and host files are dropped, they are commonly not zeroed out by the operating system or the Oracle database and therefore their space cannot be reclaimed in the thin pool. Users should take this information into account before they choose to reuse old LUNs that were once filled with files – either thick devices that may be later cloned to thin, or thin devices being reused. Prior to providing the previously used LUNs to Oracle they can be formatted (using "dd if=/dev/zero" for example on Linux/UNIX platforms). That way the old data is overwritten with zeros and later can

be reclaimed in the thin pool. Alternatively, when using Virtual Provisioning it may be better to remove previously used thin devices and create new ones, rather than reuse the old.

Oracle ASM Storage Reclamation Utility

To simplify the storage reclamation of thin pool space no longer needed by ASM objects, Oracle has developed the ASM Storage Reclamation Utility. ASRU is a Perl script that uses basic ASM commands to query the disk space, to save the current sizes within the ASM disk group for potential rollback in case of any failures, and to resize the ASM disks with rebalance to shrink the ASM disk group size. Once the ASM disks are resized, ASRU fills the remainder of the disk space with zeros to allow the reclamation of the zero space by storage Virtual Provisioning zero reclamation algorithms. The whole process is nondisruptive since users can perform ASM's resize and rebalance operations online. Follow these directions to reclaim the space ASM objects no longer need by using ASRU and Symmetrix zero space reclamation.

- Determine the size of the ASM disks associated with the specified disk group. Next, save the total used and free capacity in a private file in the disk group for rollback in the event of some failure during the execution.
- Compute the new sizes of the ASM disks, taking into account the capacity that you currently use and the capacity reserved for future growth. By default, 25 percent of the used capacity is used to calculate the reserved capacity, but you can change this figure using a command line parameter when invoking the script.
- Issue SQL commands to rebalance the ASM disk group by spreading the in-use ASM extents evenly across all ASM disks in the disk groups and resizing the disks to the lower size.
- Fill the rest of the unused portion of the ASM disks with zeros by issuing dd.
- Run Symmetrix Virtual Provisioning zero space reclamation on the thin pool to reclaim the space zeroed out by ASRU.

As described earlier, Oracle ASM reuses the freed up space. However, when a large amount of space is released after the deletion of the data file and the space is not anticipated to be needed, the ASRU utility along with Symmetrix Virtual Provisioning zero space reclamation can save a considerable amount of space. The following section describes how you can create multiple copies of Oracle data on ASM using TimeFinder even on thin devices. Later, when such copies are no longer needed, you can use ASRU to reclaim the space. An example demonstrating how you can reclaim about 57 percent of the space using Oracle ASRU with EMC Symmetrix Virtual Provisioning zero space reclamation is described in "Appendix B: An example of thin pool storage reclamation with Oracle ASRU." In this example, multiple TimeFinder/Clone copies of the source database are created to maintain a GOLD copy to start a reporting instance, or to utilize for other use cases. All the copies share the thin pool for efficiency and copies are deleted and thin pool storage reclaimed using ASRU and the EMC Virtual Provisioning zero space reclamation mechanism.

Approaches to storage replications with thin devices

Organizations can 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. Refer to the Release Notes for more details on supported configurations.

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, zeros 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.

TimeFinder/Clone supports the creation of clone sessions between regular devices and thin devices on Symmetrix VMAX arrays as of the Enginuity 5874 Q4 2009 service release. When a clone copy is made between a regular source device and a thin target device, source device tracks that are flagged as never been written to will not be copied to the target thin volume. When such a flag is not set on the source device the replication will copy all the tracks from the thick source to the thin target. This results in a fully allocated thin LUN as per the size of the source thick device.

Following the clone copy, any thin device extents that were allocated on the clone target that contain all zeros can be reclaimed and added back to the target thin device's pool. This is accomplished using Virtual Provisioning space reclamation as described earlier.

Conclusion

EMC Symmetrix Virtual Provisioning provides a simple, noninvasive, and economical way to accommodate storage requirements for Oracle databases. Oracle ASM together with Oracle database 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. The Symmetrix Enginuity 5874 Q4 2009 service release enhanced Symmetrix VMAX Virtual Provisioning with thick to thin replication, zero space reclamation, and automated pool rebalancing. Symmetrix VMAX Virtual Provisioning with recent advancements offers nondisruptive storage allocation, improved storage utilization, and ease of migration from thick to thin LUNs for reducing TCO in customer environments.

Appendix A: An example of working with Virtual Provisioning

Environment

- Symmetrix VMAX ID: 00019400191 (the scripts use "–sid 191" to refer to the Symmetrix)
- Host OS: Linux 64-bit kernel 2.6.18-92.el5
- Oracle 11g R2 using ASM
- EMC PowerPath® version 5.3.0

Setting up thin pool and devices

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

```
# symconfigure -sid 191 -cmd "create pool Oracle1 type=thin
max_subs_percent=500;" commit
```

Create 8 x 96 GB data devices and add them to the Oracle1 thin pool
The devices are created using RAID 5 (7+1) protection.

Note: The example here shows use of Solutions Enabler, but the same can be created using the Symmetrix bin file configuration change operation.

```
# symconfigure 191 -cmd "create dev count=8,size=100000 CYL, emulation=FBA,
in pool=Oracle1 member_state=ENABLE config=RAID-5, data_member_count=7,
attribute=datadev;" commit
```

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

```
...
New symdevs: 0370:0377
...
```

Create 35 x 30 GB thin devices and bind them to the Oracle1 thin pool

```
# symconfigure -sid 191 -cmd "create dev count=35,size=32000 CYL,
emulation=FBA, config=tdev binding to pool=Oracle1 ;" commit
```

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

```
...
New symdevs: 028E:2C3
...
```

Host mapping/masking of the thin devices

The Symmetrix VMAX Auto-provisioning Groups can be used to map and mask the thin devices to the database host. The task involves very simple operations like creating the initiator group, port group, and storage group and binding all three constructs together to map and mask the devices to the host automatically.

Create the initiator group containing the WWNs of the host HBA

The HBA WWNs can be included in the command line or for simplicity added to a text file for this operation. The example here uses the text file.

```
# symaccess –sid 191 create –name 44Port0_7E0 –type initiator –file WWNs.txt
```

Create the port group with the Symmetrix director and FA port number

Create a port group specifying the Symmetrix front-end port through which the storage will be visible to the host.

```
# symaccess –sid 191 create –name 44Port0_7E0 –type port –dirport 7E:0
```

Create the storage group listing the thin devices

```
# symaccess –sid 191 –name 44Port0_7E0 –type storage add devs 028E:02C3
```

Create the view binding all the above three constructs together for automatic mapping and masking of the devices through Symmetrix FA ports and host HBA ports

```
# symaccess –sid create view –name 44Port0_7E0 –sg 44Port0_7E0 –pg 44Port0_7E0 –ig
44Port0_7E0
```

List the new thin devices from the host

```
# symcfg disc
# powermt config
# sympd list
...
Symmetrix ID: 000194900191

        Device Name          Directors                   Device
-------------------------- ------------ --------------------------------------
                                                                        Cap
Physical                Sym  SA :P DA :IT  Config        Attribute    Sts   (MB)
-------------------------- ------------ --------------------------------------
/dev/emcpowereg         028E 07E:0  NA:NA  TDEV          N/Grp'd      RW   30000
/dev/emcpowereh         0341 07E:0  NA:NA  TDEV          N/Grp'd      RW   30000
/dev/emcpowerei         0340 07E:0  NA:NA  TDEV          N/Grp'd      RW   30000
/dev/emcpowerej         033F 07E:0  NA:NA  TDEV          N/Grp'd      RW   30000
….
```

View the thin pool from the host

In the output below, enabled devices are the data devices containing thin device extents. The thin pool may also contain disabled data devices not containing thin device extents. Such devices can be enabled and made available for storage allocation whenever needed.

```
# symcfg show –pool Oracle1 –sid 191 –detail –thin

Symmetrix ID: 000194900191

Symmetrix ID                  : 000194900191
Pool Name                     : Oracle1
Pool Type                     : Thin
Dev Emulation                 : FBA
Dev Configuration             : RAID-5(7+1)
Pool State                    : Enabled
# of Devices in Pool          :   8
# of Enabled Devices in Pool  :   8
Max. Subscription Percent     : 500

Enabled Devices(8):
```

```
  {
   ------------------------------------------------------
   Sym         Total      Alloc       Free Full  Device
   Dev        Tracks     Tracks     Tracks  (%)  State
   ------------------------------------------------------
   036A      1499988    1386336     113652   92  Enabled
   036B      1499988    1384800     115188   92  Enabled
…
       ---------- ---------- ---------- ----
   Tracks   11999904   11004108     995796   91
   }

Thin Devices(35):
  {
   --------------------------------------------------------------------------
                        Pool          Pool             Pool
   Sym          Total  Subs        Allocated         Written
   Dev         Tracks   (%)     Tracks   (%)     Tracks   (%)  Status
   --------------------------------------------------------------------------
   028E        480000     4     479880     4     479714     4  Bound
   028F        480000     4     480000     4     479985     4  Bound

…
          ---------- ----- ---------- ----- ---------- -----
   Tracks    16800000   140   11004108    92   11002957    92
   }
```

Adding new data devices to the thin pool

The output below shows the state of the thin pool after new data devices are added to the pool but they don't contain any data. The new data devices were earlier created using the command described earlier. These devices need to have the same configuration and protection as the existing devices in the pool. The size of the device could be different.

```
# symcfg show -pool Oracle1 -sid 191 -detail -thin

Symmetrix ID: 000194900191

Symmetrix ID                  : 000194900191
Pool Name                     : Oracle1
Pool Type                     : Thin
Dev Emulation                 : FBA
Dev Configuration             : RAID-5(7+1)
Pool State                    : Enabled
# of Devices in Pool          :   16
# of Enabled Devices in Pool  :   16
Max. Subscription Percent     : 500

Enabled Devices(8):
  {
   ------------------------------------------------------
   Sym         Total      Alloc       Free Full  Device
   Dev        Tracks     Tracks     Tracks  (%)  State
   ------------------------------------------------------
   036A      1499988    1386336     113652   92  Enabled
   036B      1499988    1384800     115188   92  Enabled
…
   0372      1499988          0    1499988    0  Enabled
   0373      1499988          0    1499988    0  Enabled
       ---------- ---------- ---------- ----
   Tracks   11999904    11096508    12903300  46
   }

Thin Devices(35):
  {
```

```
   ------------------------------------------------------------------------
                         Pool          Pool             Pool
   Sym          Total  Subs        Allocated         Written
   Dev         Tracks   (%)     Tracks   (%)     Tracks   (%)  Status
   ------------------------------------------------------------------------
   028E        480000     4     479880     4     479714     4  Bound
   028F        480000     4     480000     4     479985     4  Bound

…
            ---------- ----- ---------- ----- ---------- -----
   Tracks    16800000   140   11004108    92   11002957    92
   }
```

Rebalancing the thin pool after adding new devices

After adding new devices the thin pool can be balanced to make sure existing data extents are balanced across an all-new set of devices.

```
# symcfg –sid 191 –cmd "start balancing on pool Oracle1;" commit
# symcfg show –pool Oracle1 –sid 191 –detail –thin

Symmetrix ID: 000194900191

Symmetrix ID                  : 000194900191
Pool Name                     : Oracle1
Pool Type                     : Thin
Dev Emulation                 : FBA
Dev Configuration             : RAID-5(7+1)
Pool State                    : Balancing
# of Devices in Pool          :   16
# of Enabled Devices in Pool  :   16
Max. Subscription Percent     : 500

Enabled Devices(8):
  {
   ------------------------------------------------------
   Sym         Total      Alloc       Free Full  Device
   Dev        Tracks     Tracks     Tracks  (%)  State
   ------------------------------------------------------
   036A      1499988    1233839     101150   89  Balancing
   036B      1499988    1384800     115188   92  Enabled
   036C      1499988    1218624     101365   88  Balancing
…
   0372      1499988       1536    1498452    0  Enabled
   0373      1499988          0    1499988    0  Enabled
…
       ---------- ---------- ---------- ----
   Tracks   11999904    11096508    12903300 46
   }

Thin Devices(35):
  {
   ------------------------------------------------------------------------
                         Pool          Pool             Pool
   Sym          Total  Subs        Allocated         Written
   Dev         Tracks   (%)     Tracks   (%)     Tracks   (%)  Status
   ------------------------------------------------------------------------
   028E        480000     4     479880     4     479714     4  Bound
   028F        480000     4     480000     4     479985     4  Bound

…
            ---------- ----- ---------- ----- ---------- -----
   Tracks    16800000   140   11004108    92   11002957    92
   }
```

End of the rebalancing operation

Once the rebalancing operation completes all the data devices in the enabled state will have some extents distributed from the original set of data devices. The rebalancing operation mainly looks at the capacity utilization of the data devices and uses minimum, mean, and maximum utilization values for distribution of the extents. At the end all the devices will be within about 10 percent tolerance.

```
# symcfg show -pool Oracle1 -sid 191 -detail -thin

Symmetrix ID: 000194900191

Symmetrix ID                  : 000194900191
Pool Name                     : Oracle1
Pool Type                     : Thin
Dev Emulation                 : FBA
Dev Configuration             : RAID-5(7+1)
Pool State                    : Enabled
# of Devices in Pool          :   16
# of Enabled Devices in Pool  :   16
Max. Subscription Percent     : 500

Enabled Devices(8):
  {
   ---------------------------------------------------
   Sym         Total      Alloc       Free Full  Device
   Dev        Tracks     Tracks     Tracks  (%)  State
   ---------------------------------------------------
   036A      1499988     728160     771828   48  Enabled
   036B      1499988     726684     773304   48  Enabled
…
   0372      1499988     662028     837960   44  Enabled
   0373      1499988     662016     837972   44  Enabled
…
       ---------- ---------- ---------- ----
   Tracks   23999808   11096508   12903300   46
   }

Thin Devices(35):
  {
   ------------------------------------------------------------------------
                       Pool          Pool             Pool
   Sym          Total  Subs        Allocated         Written
   Dev         Tracks   (%)     Tracks   (%)     Tracks   (%)  Status
   ------------------------------------------------------------------------
   028E        480000     4     479880     4     479714     4  Bound
   028F        480000     4     480000     4     479985     4  Bound

…
            ---------- ----- ---------- ----- ---------- -----
   Tracks    16800000   140   11004108    92   11002957    92
   }
```

Removing data devices from the thin pool

Virtual Provisioning allows removing the devices from the thin pool nondisruptively. This can be done simply by disabling the devices from the pool. The extents allocated on the devices are automatically copied to the other enabled devices in the pool.

```
# symconfigure -sid 191 -cmd "disable dev 0342:0353 in pool Oracle1 type=thin;" commit

# symcfg show -pool Oracle1 -sid 191 -detail -thin -all

Symmetrix ID: 000194900191
```

```
Symmetrix ID                  : 000194900191
Pool Name                     : Oracle1
Pool Type                     : Thin
Dev Emulation                 : FBA
Dev Configuration             : RAID-5(7+1)
Pool State                    : Enabled
# of Devices in Pool          :   40
# of Enabled Devices in Pool  :   22
Max. Subscription Percent     : 500

Enabled Devices(8):
  {
   ----------------------------------------------------
   Sym          Total      Alloc       Free Full  Device
   Dev         Tracks     Tracks     Tracks  (%)  State
   ----------------------------------------------------
   036A      1499988    1386336     113652   92  Enabled
   036B      1499988    1384800     115188   92  Enabled
…
   0372      1499988          0    1499988    0  Enabled
   0373      1499988          0    1499988    0  Enabled
       ---------- ---------- ---------- ----
   Tracks   11999904    11096508     12903300  46
   }

Disabled Devices(18):
  {
   ----------------------------------------------------
   Sym          Total      Alloc       Free Full  Device
   Dev         Tracks     Tracks     Tracks  (%)  State
   ----------------------------------------------------
   0342       491520          0     491520    0  Disabled
   0343       491520          0     491520    0  Disabled
…
   0352       491520     354048     137472   72  Draining
   0353       491520     351336     140184   71  Draining
   }

Thin Devices(35):
  {
   ------------------------------------------------------------------------
                        Pool          Pool              Pool
   Sym           Total  Subs        Allocated         Written
   Dev          Tracks   (%)     Tracks   (%)     Tracks   (%)  Status
   ------------------------------------------------------------------------
   028E         480000     4     479880     4     479714     4  Bound
   028F         480000     4     480000     4     479985     4  Bound

…
            ---------- ----- ---------- ----- ---------- -----
   Tracks    16800000   140   11004108    92   11002957    92
   }
```

Appendix B: An example of thin pool storage reclamation with Oracle ASRU

Environment

- Symmetrix VMAX ID: 00019400191 (the scripts use "–sid 191" to refer to the Symmetrix)
- Host OS: Linux 64-bit kernel 2.6.18-92.el5
- Oracle 11g R2 using ASM
- EMC PowerPath® version 5.3.0

Setting up a thin pool and devices and creating Oracle database on ASM using the thin pool

The thin pool Oracle1 is used for the source Oracle ASM instance using the mechanism described in Appendix A. PE_DATA1, PE_FRA1, PE_REDO1, and PE_TEMP1 ASM disk groups are created to store Oracle data as per the Symmetrix best practices described earlier. The thin pool is created on 26 x 240 GB data devices with the total available capacity of 6.3 TB. An EMC IT Enterprise Data Warehouse source database of 1 TB in size is created on 59 x 30 GB thin devices with enough capacity for expansion.

ASM disk group layout and thin pool storage allocation

PE_DATA1: 50 thin devices

PE_FRA1: 3 thin devices

PE_REDO1: 3 thin devices

PE_TEMP1: 3 thin devices

After the database is created, ASM disk group and ASM disk storage allocation are reported below. The PE_DATA1 disk group capacity is 1.5 TB and only 5 GB is free.

```
# State of ASM DGs and disks before running ASRU
SQL> select name, total_mb, free_mb from v$asm_diskgroup;

NAME                                TOTAL_MB     FREE_MB
------------------------------ ---------- ----------
PE_DATA1                           1499950        5943
PE_FRA1                             89997      89941
PE_REDO1                            89997      89941
PE_TEMP1                            89997      89941

SQL> select name, path, total_mb, free_mb from v$asm_disk;

NAME
-----------------------------
PATH
--------------------------------------------------------------------------------
  TOTAL_MB    FREE_MB
---------- ----------
PE_TEMP1_0000
/dev/emcpowerhw1
     29999      29981

PE_DATA1_0024
/dev/emcpowerju1
     29999        122

...

PE_REDO1_0002
/dev/emcpowerhv1
     29999      29982

...

PE_FRA1_0001
/dev/emcpoweria1
     29999      29982
...
```

The storage allocation in thin pool Oracle1 after the source database was created is as follows:

```
# symcfg show -pool Oracle1 -sid 191 -detail -thin -all

# Pool prior to creating itdb1 clone

Symmetrix ID: 000194900191

Symmetrix ID                  : 000194900191
Pool Name                     : Oracle1
Pool Type                     : Thin
Dev Emulation                 : FBA
Dev Configuration             : RAID-5(3+1)
Pool State                    : Enabled
# of Devices in Pool          :  26
# of Enabled Devices in Pool  :  26
Max. Subscription Percent     : None

Enabled Devices(26):
  {
   ----------------------------------------------------------
   Sym         Total      Alloc       Free Full  Device
   Dev        Tracks     Tracks     Tracks  (%)  State
   ----------------------------------------------------------
   009D      3939984         12    3939972    0  Enabled
...
```

```
   028E      3939984    1991352    1948632   50  Enabled
   028F      3939984    1989960    1950024   50  Enabled
...
        ---------- ---------- ---------- ----
   Tracks  102439584   23892204   78547380   23
   }

Thin Devices(126):
  {
   --------------------------------------------------------------------------
                        Pool          Pool             Pool
   Sym           Total  Subs        Allocated         Written
   Dev          Tracks   (%)     Tracks   (%)     Tracks   (%)  Status
   --------------------------------------------------------------------------
   00EA         480000     0         12     0          0     0  Bound

...
   025D         480000     0     478344     0     478321     0  Bound
   025E         480000     0     478164     0     478137     0  Bound

...
         ---------- ----- ---------- ----- ---------- -----
   Tracks    60480000    59   23892204    23   23884391    23
  }
```

Based on these results, the Oracle source database used 1.5 TB allocated capacity on the Oracle1 thin pool and most of this capacity was occupied by the database. This shows the effect of the Oracle database file initialization that results in full allocation of data file space in the thin pool.

Adding thin devices to the pool for a thin LUN-based clone of the source database

The thin pool can be shared by multiple ASM instances and applications for efficiency. In this example, new sets of thin devices are added to the thin pool Oracle1 and used as the targets of TimeFinder/Clone of the source database. The TimeFinder/Clone session is created and activated between source and target device pairs. Two sets of TimeFinder/Clone are created on the thin pool for different uses.

```
# symcg create itdb1_c1
# symcg -cg itdb1_c1 -sid 191 addall dev -RANGE 025D:02BF
# symcg -cg itdb1_c1 -sid 191 addall dev -tgt -RANGE 00EA:0128
# symcg -cg itdb1_c1 create -diff -precopy

The state of the pool after clone created and copied.
# symcfg show -pool Oracle1 -sid 191 -detail -thin -all

Symmetrix ID: 000194900191

Symmetrix ID                  : 000194900191
Pool Name                     : Oracle1
Pool Type                     : Thin
Dev Emulation                 : FBA
Dev Configuration             : RAID-5(7+1)
Pool State                    : Enabled
# of Devices in Pool          :   26
# of Enabled Devices in Pool  :   26
Max. Subscription Percent     : 500

Enabled Devices(26):
  {
   --------------------------------------------------------
   Sym          Total      Alloc       Free Full  Device
   Dev         Tracks     Tracks     Tracks  (%)  State
   --------------------------------------------------------
   028E       3939984    2931984    1008000   74  Enabled
```

```
   028F      3939984    2907756    1032228   73  Enabled
...
       ---------- ---------- ---------- ----
   Tracks    102439584    47783916    54655668    46
   }

Thin Devices(189):
  {
   --------------------------------------------------------------------------
                       Pool          Pool             Pool
   Sym          Total  Subs        Allocated         Written
   Dev         Tracks   (%)     Tracks   (%)     Tracks   (%)  Status
   --------------------------------------------------------------------------
   028E        480000     4     479880     4     479714     4  Bound
   028F        480000     4     480000     4     479985     4  Bound

…
           ---------- ----- ---------- ----- ---------- -----
   Tracks     90720000     89    47783916     47    47769102     47
   }
```

Calculating the thin pool storage occupied by the first set of clones

As we can see from the previous reports, the thin pool storage allocation by the source database was 23,892,204 tracks (1.5 TB) and the total storage allocated after the clone was 47,783,916 (3.1 TB). The TimeFinder/Clone operation between the thin source and target devices only copies allocated tracks and hence each additional copy will result in the same amount of thin pool allocation.

Database	Allocated Tracks	Allocated Capacity	Total Allocated Pool Capacity
Source: itdb1	23,892,204	1.56 TB	1.56 TB
Clone 1: itdb1_c1	47,783,916	1.56 TB	3.13 TB

Dropping the TimeFinder/Clone of the Oracle database that is no longer needed and running ASRU

The TimeFinder/Clone copy used to start an Oracle database instance for reporting is no longer needed. This clone Oracle database is now dropped using the following SQL command.

```
Shutdown the database and start it up again in restrict mode to drop the database

# sqlplus "/ as sysdba"
SQL> shutdown immediate

SQL> startup mount exclusive restrict
SQL> drop database;
```

The ASM disk group will immediately show the free space that can be reused for subsequent uses and the thin pool space will remain allocated by the ASM disk groups.

```
SQL> select name, total_mb, free_mb from v$asm_diskgroup;

NAME                            TOTAL_MB    FREE_MB
------------------------------ ---------- ----------
PE_DATA1                          1499950    1498450
```

```
PE_FRA1                          89997      89941
PE_REDO1                         89997      89941
PE_TEMP1                         89997      89941
```

If the space allocated by the disk group is no longer needed by the Oracle-provided ASRU utility, you can use the space to resize the ASM disks and zero out the rest of the disk.

```
The current ASM disk sizes are shown below:
SQL> select name, path, total_mb, free_mb from v$asm_disk;

NAME
------------------------------
PATH
--------------------------------------------------------------------------------
  TOTAL_MB    FREE_MB
---------- ----------
PE_DATA1_0024
/dev/emcpowerju1
     29999      29971


Running ASRU on all ASM disk groups.

# ASRU PE_DATA1
# ASRU PE_REDO1
# ASRU PE_FRA1
# ASRU PE_TEMP1
```

ASRU resizes the disk to a lower high water mark and zeros out the rest of the disk up to the total size reported here, which in this case is 30 GB. After ASRU runs, the disk size is reduced.

Gathering ASRU disk resize information

The ASRU utility reports the total sizes and new sizes of all ASM disks in the disk group. It creates a temporary file in the ASM disk group with the information about the actual size of the disk and then resizes the disks. The sample output is shown below:

```
PE_DATA1_0001: 29999 29969 30 38
PE_DATA1_0002: 29999 29969 30 38
...

Writing the data to a file ...

Data to be recorded in the tp file : PE_DATA1_0001 29999 PE_DATA1_0002 29999 ...
Resizing the disks...

Executing ALTER DISKGROUP PE_DATA1 RESIZE DISK PE_DATA1_0001 SIZE 37M DISK PE_DATA2 SIZE
37M ... REBALANCE WAIT

Retrieving the paths:
PE_DATA1_0001: /dev/emcpowerjre1
PE_DATA1_0002: /dev/emcpowerjre2

...

Executing /bin/dd if=/dev/zero of=/dev/emcpowerjre1 seek=39 bs=1024k count=29960 ...
```

When the ASRU utility successfully finishes the ASM disks are resized to 37M and the dd command is run to fill the disk with zero, starting with the new offset all the way to the end of the disk.

Running Symmetrix zero space reclamation on the thin pool

At this point the disks are resized and filled with zeros. Symmetrix thin pool space reclamation is now run to reclaim the contiguous zero space in the pool.

```
# symconfigure –sid 191 –cmd "free tdev 00EA:0128 start_cyl=0 size=32000 CYL
type=reclaim; " commit

The state of the pool after reclamation is done
# symcfg show –pool Oracle1 –sid 191 –detail –thin -all

Symmetrix ID: 000194900191

Symmetrix ID                  : 000194900191
Pool Name                     : Oracle1
Pool Type                     : Thin
Dev Emulation                 : FBA
Dev Configuration             : RAID-5(7+1)
Pool State                    : Enabled
# of Devices in Pool          :   26
# of Enabled Devices in Pool  :   26
Max. Subscription Percent     : 500

Enabled Devices(26):
  {
   ----------------------------------------------------
   Sym         Total      Alloc       Free Full  Device
   Dev        Tracks     Tracks     Tracks  (%)  State
   ----------------------------------------------------
   009D      3939984     516732    3423252   13  Enabled
   009E      3939984     515784    3424200   13  Enabled
          ---------- ---------- ---------- ----
    Tracks  102439584   37440720   64998864   36
}

Thin Devices(189):
  {
   --------------------------------------------------------------------------
                        Pool          Pool              Pool
   Sym          Total  Subs        Allocated         Written
   Dev         Tracks   (%)     Tracks   (%)     Tracks   (%)  Status
   --------------------------------------------------------------------------
   028E        480000     4     479880     4     479714     4  Bound
   028F        480000     4     480000     4     479985     4  Bound

…
        ---------- ----- ---------- ----- ---------- -----
   Tracks    90720000    89   37440720    37   37427799    37   }
```

The thin pool (oracle1) consumption changed as described below:

With the source database only: **23892204 (1.56 TB)**
With the source database and TimeFinder/Clone of the source database: **47783916 (3.13 TB)**

Storage allocated and consumed by the TimeFinder/Clone in the thin pool: **23891712 (1.56 TB)**

The thin pool after running ASRU on unused ASM storage used by TimeFinder/Clone and reclaiming space using Symmetrix zero space reclamation: **37440720 (2.45 TB)**

Space reclamation with ASRU in conjunction with zero space reclamation: **10343196 (631 GB)**

The percentage of space reclamation after ASRU and Symmetrix zero space reclamation on the thin pool was **57%.**

This shows that ASRU in conjunction with Symmetrix zero space reclamation results in reclamation of 57 percent of the allocated space. Note that ASRU reserves 25 percent of the space for future use and the actual reclamation and hence space savings can be even higher if this default is changed using a command line argument.

Appendix C: Aligning 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
```

Creating an ASM disk group 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
$
```