White Paper

EMC CLARiiON MetaLUNs

A Detailed Review

Abstract

This white paper describes the terminology, concepts, O/S support, storage-system operations, and management of EMC® CLARiiON® MetaLUNs. MetaLUNs are typically used to expand the capacity of an existing LUN, but may be used to create very large-capacity or very high-performance LUNs.

October 2011

EMC²

Part Number H1024.3

Table of Contents

Executive summary

MetaLUNs are an EMC® CLARiiON® standard feature available on all models as a part of the CLARiiON storage system's FLARE® operating environment. A MetaLUN is two or more joined FLARE Logical Units (LUNs) acting in concert, presented to a host as a single LUN. MetaLUNs permit FLARE LUNs to be scaled upward when the performance or capacity requirement exceeds that available from the maximum number of drives in a LUN bound to a single RAID group. They also allow users to provision their CLARiiON to:

- Expand the capacity of existing LUNs "on the fly" while maintaining performance
- Create LUNs with very large capacity
- Create higher-performance LUNs

You need to plan carefully to provision for optimal capacity, performance, availability, and ease of expansion. This white paper provides the information you need to use CLARiiON's MetaLUNs to make your CLARiiON storage environment even more cost-effective.

Introduction

MetaLUNs are an important, standard feature in CLARiiON. MetaLUNs can be used to increase the capacity of existing LUNs, create LUNs with a larger capacity than a FLARE LUN, increase the performance of a LUN, create LUNs with greater performance than a FLARE LUN, and in some cases increase the availability of a LUN. While not an advanced feature, creating and maintaining MetaLUNs requires additional planning and effort.

This white paper contains both background and MetaLUN-specific information. General information is found in the beginning of the paper. Readers who already have an in-depth understanding of RAID groups and LUNs may skip those sections and go directly to the MetaLUN-specific sections for information on how to configure and use MetaLUNs for optimal performance and availability.

The paper is divided into the following major topics of discussion:

- Overview: This provides an overview and terminology
- Meta LUN concepts: FLARE LUNs, simple and complex MetaLUNs
- MetaLUN provisioning: Capacity, performance, and availability
- MetaLUN alternatives: Other capacity and performance provisioning options, including storage virtualization
- MetaLUN management: Applications used to create, modify, and destroy MetaLUNs

For specific information on the current FLARE version's usage of and recommendations regarding MetaLUNs, refer to the white paper *EMC CLARiiON Best Practices for Performance and Availability*. This document is referred to as Best Practices. This paper is revised with each revision of FLARE. It contains the most current thought on MetaLUN performance and availability.

Users wanting more background information on FLARE, CLARiiON provisioning, LUNs, and RAID groups should refer to the white paper *EMC CLARiiON Storage System Fundamentals*. This document is referred to as Fundamentals.

Both of these documents are available on Powerlink®.

Audience

This white paper is for the use of customers, including storage architects and administrators, and any others involved in evaluating, acquiring, managing, operating, or designing an EMC networked storage environment. In addition, EMC staff and partners may refer to it for guidance and development of proposals.

MetaLUNs terminology and overview

Following is a summary of terms and concepts found in this paper. Both old and current terms are included. Older terms are included as a reference, while current terms are used throughout this paper.

Terminology

Alignment — How data is laid out on the set of a RAID group's stripe elements; organizing alignment allows you to minimize disk crossings.

Base component — The original MetaLUN used in the expansion of an existing MetaLUN. The base component is a MetaLUN component sometimes called Component 0.

Base LUN — The founding FLARE LUN from which the MetaLUN is created. The MetaLUN is created by expanding the base LUN's capacity. The base LUN is the only LUN whose data and attributes are preserved within the newly created MetaLUN.

Complex MetaLUN — A MetaLUN whose component LUNs are joined by both striped and concatenated expansion methods.

Component LUN — A member of the set of FLARE LUNs joined into a MetaLUN component. Component LUNs are logical storage objects. A component is either a single LUN previously concatenated or several component LUNs striped within an existing MetaLUN. The base LUN is also a component LUN.

Concatenated expansion —MetaLUN expansion method appending new components to the base LUN or base component. This type of expansion adds storage in a serial way.

Dedicated RAID group — A RAID group whose entire capacity is bound into one LUN.

Destroy — To take apart a MetaLUN and return its resources for reuse.

Dirty LUN — (archaic) See In-use LUN.

Disk Crossing —An I/O operation that requires reading and/or writing from two drives instead of one.

Exact bind — The creation or expansion of a MetaLUN to a usable capacity that is less than the summed capacity of the component LUNs.

Expansion — Joining a component to a base FLARE LUN component or a base MetaLUN component. Base FLARE LUN components may only be expanded with FLARE LUN components.

FLARE LUN — A logical storage object created by partitioning a CLARiiON RAID group.

FLU — See FLARE LUN.

Heterogeneous component LUNs —The base LUN and the component LUNs have different LUN capacities or underlying RAID group organization.

Homogenous component LUNs —The base LUN and the component LUNs have the same LUN capacities and underlying RAID group organization.

In-use LUN — A LUN that has been marked within the storage system software as in use. It is assumed that these LUNs have live data on them. For example, a LUN that is in a storage group, labeled by a host, and then removed from the storage group is a dirty LUN. This is because although the LUN is not in a storage group, it still contains data.

I/O Contention — When the I/Os to LUNs bound to the same RAID group interfere with each other's performance.

Logical Volume Manager (LVM) — A host-based storage virtualization technology.

Level of protection — The number of failures a RAID group can sustain before losing data.

Metadata — Information about a data set embedded in the data set or storage object.

MetaLUN — A LUN consisting of two or more component LUNs joined together whose summed capacity is presented to a host or hosts.

MetaLUN component — A logical storage object consisting of one or more FLARE LUNs that have been concatenated onto another MetaLUN component.

Owner — The storage processor responsible for handling the I/O to a particular LUN.

Partitioned RAID group — A RAID group whose capacity is bound into two or more LUNs. These LUNs may be owned by the same or peer storage processors.

Private LUN — A LUN managed by FLARE and not directly addressable by a host.

Scalable — When capacity and performance can easily be increased to meet growing needs.

Simple MetaLUN — A MetaLUN whose component LUNs are joined by a single expansion method.

Stacked MetaLUN — A MetaLUN with two or more component LUNs bound on the same RAID group. Sometimes referred to as a Vertically Striped MetaLUN.

Striped expansion — A type of MetaLUN expansion that stripes the data of the base component across both the original and new components being added. This type of expansion adds storage in a parallel way.

Total capacity — The summed capacity of all the FLARE LUNs that make up a MetaLUN. This includes capacity available to the user and metadata.

User capacity — The capacity of a MetaLUN that is presented to a host. User capacity is set by the user and is less than the total capacity.

Vertically striped MetaLUN – See Stacked MetaLUN.

Virtual Provisioning™ — A storage system-based storage virtualization feature.

Overview

MetaLUNs allow for provisioning larger and higher-performance LUNs. The simplest use of MetaLUNs is to expand a LUN's capacity while maintaining its level of performance. You do this by joining a newly created LUN to an existing LUN that no longer has any available capacity or needs. The new "larger" LUN appears to a host to be identical to the original, except now with more capacity.

When you create a MetaLUN, you join a base LUN to component LUNs. The base LUN and component LUNs originally are FLARE LUNs. A MetaLUN can be scaled in a step-wise process by joining additional FLARE LUNs to it to make a larger MetaLUN. A simple MetaLUN may be made up of wholly component LUNs. A complex MetaLUN may be made up of MetaLUN components, where each MetaLUN component is one or more component LUNs. MetaLUNs may be homogenous or heterogeneous in organization. That is, the FLARE LUNs joined into a MetaLUN may have the same underlying RAID group organization or they may come from RAID groups with different RAID levels with compatible redundancy schemes, and varying numbers of drives.

The maximum possible capacity of a FLARE LUN is the maximum number of drives in a RAID group (16). For example, the largest recommended capacity FLARE LUN using available 2 TB SATA drives and a 12-drive RAID 6 would have a usable capacity of about 18.3 TB. Using MetaLUNs allows two such LUNs to be joined together to create a larger 36.6 TB host LUN. More LUNs can be joined to make even larger capacity MetaLUNs. The maximum capacity of a MetaLUN is only limited by the number of drives available on the storage system. Another benefit of creating LUNs with more drives is increased performance. Using the correct techniques, MetaLUNs usage can increase the throughput (IOPS) and bandwidth of an existing LUN or create LUNs with higher IOPS and bandwidth than may be available with a single FLARE LUN.

The initial provisioning and maintaining of MetaLUNs is in addition to FLARE LUN creation and maintenance. Understanding the fundamentals of FLARE LUNs and their underlying RAID group's organization is important if you wish to make use of the full

potential of MetaLUNs. In addition, there are several alternative methods on the CLARiiON for expanding the capacity or increasing the performance of a LUN. For example, LUN migration can create additional LUN capacity, Storage virtualization is another possibility. CLARiiON Virtual Provisioning can create LUNs with built-in expansion capability. Host Logical Volume Managers are another virtualization alternative. In some cases, these alternatives may be more straightforward to implement or, in the long run, easier to maintain than MetaLUNs.

Basic LUN concepts

A LUN is *scalable*, when its capacity and performance can easily be increased to meet growing needs. CLARiiON uses MetaLUNs to scale FLARE LUNs upward in capacity and performance. A brief overview of FLARE LUNs are described below; if you are already have a basic understanding or FLARE LUNs and MetaLUNs you may wish to skip these sections.

FLARE LUNs

A FLARE LUN is a logical construct overlaid onto CLARiiON RAID groups. Hosts see LUNs as physical drives. LUNs are frequently referred to as disks, *volumes*, or *partitions* depending on the context. LUNs hide the organization and composition of RAID groups from hosts. LUNs are created to allocate capacity and ensure performance, and for information security. The process of creating a LUN is called binding, with LUNs eventually being bound to a RAID group at the end of the process.

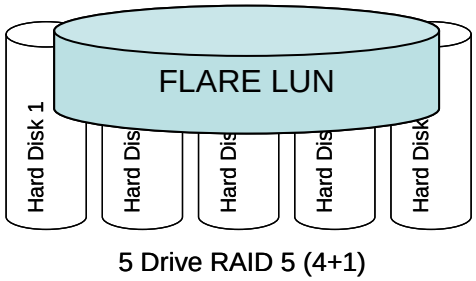


Figure 1 FLARE Logical Unit (LUN) conceptual view

Hosts access LUNs through CLARiiON *storage groups*. A storage group controls access to reading and writing data, configuration changes, and the management of storage system resources by partitioning LUNs from access by all hosts.

The number of FLARE LUNs that can be assigned to a storage group is CLARiiON model and FLARE revision dependent.

Table 1 Maximum number of FLARE LUNs

FLARE REVISION	CX4-120	CX4-240	CX4-480	CX4-960
Maximum LUNs FLARE 29.0 and later	1024	2048	4096	8192
Maximum LUNs FLARE 28.0	1024	1024	4096	4096

MetaLUNs

A MetaLUN is a logical storage object. A MetaLUN is made up of underlying component LUNs; a MetaLUN is a logical construct overlaid onto two or more component LUNs. A MetaLUN consists of multiple LUNs that are presented to a host as a single LUN. The process of creating a MetaLUN requires an understanding of CLARiiON RAID groups and FLARE LUNs.

The simplest type of MetaLUN is created from FLARE LUNs. MetaLUNs of FLARE LUNs are created using a single expansion. After the expansion the ex-FLARE LUNs are referred to as *component* LUNs. Larger capacity and higher performance MetaLUNs use two types of expansion. They are discussed in the "Complex MetaLUNs" section.

RAID groups and LUNs

A FLARE LUN's capacity, performance, and availability all come from the underlying RAID group. The type and number of drives in the RAID group and its RAID level define its capacity, performance, and availability.

Depending on the storage system and the FLARE revision, the CLARiiON supports the following drive types: Fibre Channel, SAS, SATA, and Enterprise Flash Drive (EFD). The more drives in a RAID group the higher its capacity and performance will be. The maximum number of drives in a RAID group is 16. The minimum number is RAID level dependent.

CLARiiON RAID groups offer two main types of data protection: parity and mirrored. Parity and mirror RAID levels differ in how they handle data redundancy to provide availability in the event of a drive failure. The CLARiiON's parity RAID levels are 3, 5, and 6. The mirrored RAID levels are 1 and 1/0. In addition, the CLARiiON supports RAID 0.

Table 2 RAID level levels-of-protection

RAID level	Level of protection
0	0*
1	1
3	1
5	1
6	2
1/0	1*

Table 2 shows the relationship between RAID level and *level of protection*. A single drive protected (a "1" in the table) RAID level provides data protection against a single drive failure in a RAID group. Parity RAID level 6 is a *double drive protect*ed RAID level. It provides data protection against two simultaneous RAID group drive failures.

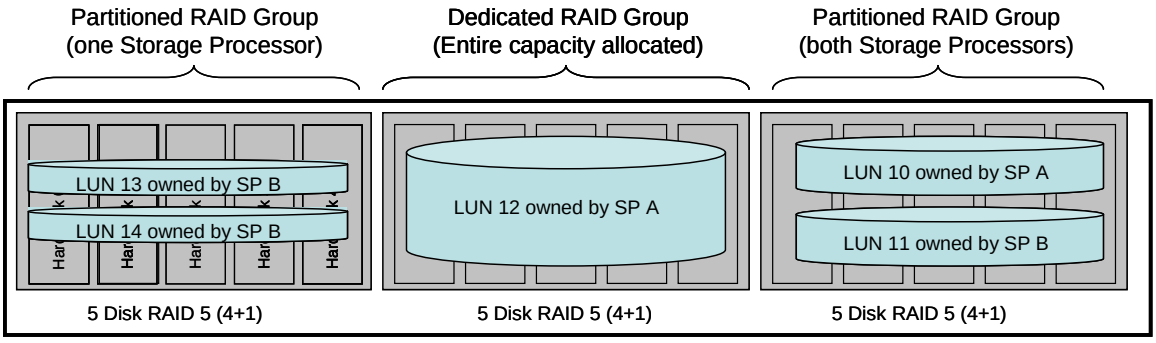
The CLARiiON's RAID level 0 and 1/0 are special cases. RAID 0 is an unprotected RAID level; it does not provide any data protection should a RAID 0 group drive fail. RAID 1/0 has a single drive level of protection. Under certain circumstances two or more drives can fail with no data being lost. This multiple failure condition is survivable only if the failing drives are not mirrored pairs. However, for practical purposes RAID 1/0 is a single drive failure level of protection.

Matching a FLARE LUN's I/O to its underlying RAID group's stripe size is an important performance optimization. Stripe size is the amount of user data in a RAID group stripe. This does not include drives used for parity or mirroring. The stripe is measured in KB. It is calculated by multiplying the number of stripe drives by the stripe element size. The default stripe element size for the CLARiiON is 64 KB. Stripe element size is not changeable.

Related to stripe size is stripe width. Stripe width is the number of hard drives in a RAID group stripe. For example, a five-drive RAID level 5 group (4+1) has a stripe width of four. An eight-drive RAID level 1/0 group (4+4) has a stripe width of four.

Partitioning RAID groups with LUNs

A RAID group can be partitioned into one or more LUNs. A FLARE LUN can be of any size capacity-wise from one block to the maximum usable capacity of the RAID group. A RAID group with only one LUN taking up all the available user capacity is a *dedicated* RAID group. A RAID group having more than one LUN is a *partitioned* RAID group. The maximum number of FLARE LUNs per RAID group is FLARE version dependent. For FLARE revision 29.0 and later the maximum number of LUNs per RAID group is 256.



15 Drive CLARiiON DAE provisioned with three 4+1 RAID groups

Figure 2 Dedicated and partitioned RAID groups

A LUN created with a dedicated RAID group has the maximum possible capacity. It also has the most predictable performance.

LUNs created from partitioned RAID groups share both the capacity and the performance of the RAID group among the partitioning LUNs. The capacity allocation is straightforward. The performance implications can become complex. This is particularly true when a RAID group is partitioned into a large number of LUNs. These LUNs all have equal access to the underlying RAID group's IOPS and bandwidth. For this reason, if separate applications use LUNs created on the same RAID group, their combined usage needs may exceed the RAID group's performance capabilities.

Contention is when the I/O of LUNs bound to the same RAID group interfere with each other's performance. In a partitioned RAID group it is important to minimize *linked contention* and *drive contention*. Linked contention is when I/Os to different LUNs force the RAID group's drive heads into large movements going from LUN to LUN. Small-block random I/O does not cause linked contention. However, a badly partitioned RAID group may experience degraded performance with any sequential I/O.

Drive contention is when more than one I/O stream needs to access the same drive at the same time. As the number of LUNs partitioning a RAID group increases it becomes more difficult to predict or determine if the I/O is complementary or contentious without time-consuming analysis and debug.

Note that a partitioned RAID group can contain LUNs owned by one of the storage system's storage processors (SPs), or both of the storage system's SPs. When a RAID group is partitioned between both SPs, this is called dual-ownership. There is the potential for more I/O optimization to be performed at the drive level with dual ownership. More IOPS may be gotten out of RAID groups with dual ownership. This results in higher utilization. However, typically LUN response times increase with dual ownership. The underlying combined read and write loads on all member LUNs on a dually-owned RAID group need to be balanced; I/O workload needs to be expected from two sources (both SPs). It is possible that one SP can monopolize the RAID group, at least temporarily.

Creating a LUN

The process of creating a LUN is called binding. When a FLARE LUN is bound it is assigned attributes that define it and identify it. These attributes include:

- Array LUN ID
- Host LUN ID (if it is in a storage group, the HLU is assigned after the LUN is bound and added to a storage group)
- WWN (UID)
- RAID group ID and RAID type
- Bind time stamp
- LUN name (assigned a LUN name after the LUN is bound)

- Any pre-existing SCSI info such as Reservations (set by the host after a LUN in bound and assigned to a storage group) and NACA settings
- LUN ownership and trespass attributes such as default owner and auto-assign

Private LUNs

When a FLARE LUN is bound, it becomes available for discovery and assignment to a host. Some storage system LUNs cannot be assigned to hosts. A private LUN is a LUN managed by FLARE and not directly addressable by a host. Private LUNs support user-related storage objects such as clones, MetaLUN components, and hot spare LUNs. Private LUNs are assigned special LUN IDs. Private LUN IDs are separate from host usable FLARE LUN IDs. Users have no control over private LUNs or private LUN ID assignment.

The maximum number of private LUNs is FLARE revision- and model-dependent. The number of possible private LUNs is part of the same maximum for FLARE LUNs. That is, if there is a maximum of 1,024 FLARE LUNs on the storage system, this maximum has to be partitioned between FLARE LUNs and private LUNs. An example is 960 FLARE LUNs and 64 private LUNs (1024 = 960 +64).

Simple MetaLUNs

The great majority of MetaLUNs in use are simple MetaLUNs, which is a MetaLUN whose component LUNs are joined by a single expansion method. A complex MetaLUN may consist of one or more MetaLUN components that are made up of one or more component LUNs. The higher capacity and performance alternatives are discussed in the “Complex MetaLUNs” section.

MetaLUN implementation

A stack describes the hierarchy of software applications performing related functions. MetaLUNs are implemented as part of the CLARiiON’s software stack. A description of the CLARiiON’s software stack can be found in the EMC CLARiiON *Storage System Fundamentals* paper. The MetaLUN software driver is located very low on the stack, just above the core FLARE operating environment, and below the replication layered applications such as SnapView™, MirrorView™, and SAN Copy™. This positioning results in very little performance overhead for the MetaLUN storage object. In addition, MetaLUNs appear as a LUN logical storage object to replication layered applications and host-based applications.

Components

A component is a logical storage object. A component contains one or more component LUNs. A simple MetaLUN can consist of one or up to 16 components. A single component can contain from two to 32 component LUNs. The component LUNs within each MetaLUN component are by default striped.

Component LUNs

A simple MetaLUN consists of two or more logical storage objects called component LUNs. Simple MetaLUNs use a single method of expansion. More complex MetaLUNs can be created through a combination of both expansion types. The expansion method used determines how many MetaLUN components comprise a MetaLUN.

In single-expansion type MetaLUNs, the MetaLUN component is made up of one or more component LUNs. Striped expansion allows component LUNs to have between two and 32 FLARE LUNs. Concatenated expansion initially restricts a MetaLUN component to a single component LUN. On the CX4 series CLARiiON storage systems there can be a maximum of 16 components concatenated within a MetaLUN. Note that in this case these 16 components would each consist of a single component LUN. (The component LUN by itself is a MetaLUN component.)

Component LUNs (which include the base LUN) are FLARE LUNs that are converted to private LUNs by FLARE after an expansion has been performed. The maximum number of FLARE LUN IDs available can determine how many component LUNs may be created at any one time. Note that FLARE LUNs are typically in use and that other CLARiiON features use private LUNs, which consume LUN IDs in their operation (see the "Private LUNs" section). The available number of LUN IDs may restrict the number and size of the MetaLUNs that may be created on smaller-model CLARiiONs.

Component LUNs cannot be private LUNs, or transitioning to private LUNs. Component LUNs cannot be in use by CLARiiON feature applications, such as SnapView snapshots, MirrorView clones, or MirrorView or SAN Copy sessions.

Certain restrictions apply to the number of component LUNs that may be expanded on legacy CLARiiONs and FLARE revisions earlier than 28.0. For specific information on legacy CLARiiON MetaLUNs (CX3 and CX series), please consult the storage system's documentation.

Expansion

Expansion is the process of creating a MetaLUN storage object. An expansion is similar to a FLARE LUN bind. The component LUNs are joined to the base LUN to create a larger logical storage object. Expansion does not create a new physical storage object. To a host, this new logical storage object (the MetaLUN) appears as the original base LUN, except with additional storage capacity. All or a portion of the additional capacity may be used in the MetaLUN. This additional capacity appears as a continuous address space starting at the base LUN's first logical block address. Base components may be newly bound LUNs with no data or have in-use capacity. A base LUN's in-use capacity is preserved during the expansion.

The number of MetaLUNs that may be created on a CLARiiON is dependent on the CLARiiON model, and secondarily on the storage system's FLARE operating system revision.

Component LUNs of a MetaLUN cannot be moved after an expansion. That is, you cannot unbind individual component LUNs or use LUN migration on the individual

LUNs of a MetaLUN to relocate them within the storage system's RAID groups. Before starting an expansion, plan ahead to ensure the correct provisioning of the component LUNs,

There are two ways to expand a MetaLUN, striped and concatenated. Both methods of expansion increase capacity. The two methods differ in the way user data is stored on their component LUNs. The different methods of storage affect the options for the provisioning of the expansion, performance, and availability. In addition, the capacity of either a striped or concatenated expansion may be adjusted downward to meet specific capacity needs at the time that the MetaLUN is created.

Base LUN

A MetaLUN is made up of a *base* LUN and component LUNs. Both base and component LUNs were originally FLARE LUNs.

The base LUN is always located in the *base component*. The base component is sometimes called component 0. The base LUN is a special component LUN that provides the MetaLUN's identity (LUN ID) for addressing purposes. The LUN ID (sometimes called the LUN *address*) of the base LUN that is expanded becomes the LUN ID for the new MetaLUN. The LUN IDs of the other FLARE LUNs in the MetaLUN expansion are reassigned as private LUNs and their original LUN ID addresses become available for re-use.

Note that there are a limited number LUN IDs per storage system. The number is FLARE revision- and model-dependent. In rare cases, the availability of LUN IDs may affect the size and maximum number of MetaLUNs that can be created.

The LUN Name and LUN ID are separate properties of a LUN. The LUN Name is not used to track the LUN internally by FLARE. The LUN Name is a text field that defaults to "LUN XXX" where XXX is the FLARE LUN ID. Note that when you create a MetaLUN, FLARE changes the LUN IDs in the background. It does not change the LUN Name to match the LUN ID. It may be necessary to manually change the Name fields of the component LUNs to reflect their MetaLUN membership.

A MetaLUN's capacity is the combined capacities of the base LUN and all of the component LUNs that make up the MetaLUN. Additional LUNs can be added to a MetaLUN to further increase its capacity. During MetaLUN expansion, the base LUN's data is always accessible. The base LUN is the only LUN whose data is preserved at the end of the expansion operation. The additional capacity is not available until the expansion completes.

Component FLARE LUNs

Component LUNs are FLARE LUNs that are joined to the base LUN to create the MetaLUN. The maximum number of FLARE LUNs in a component is 32 (base LUN + 31 component LUNs).

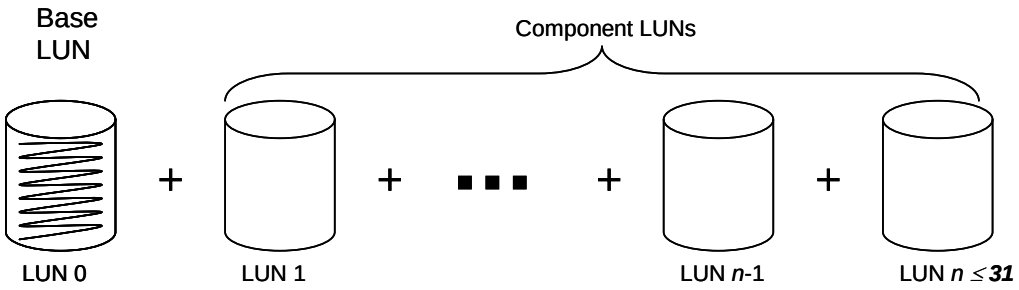


Figure 3 Striped MetaLUN: Base LUN and component LUNs

Figure 3 shows the relationship of the base LUN to the component LUNs in a striped MetaLUN. Note that striped MetaLUNs can have an 1:n (where n ☐ 32) relationship between MetaLUN components and component LUNs. Component LUNs may be striped together in groups with a single operation. That is, striping sweeps all component LUNs into a single MetaLUN component.

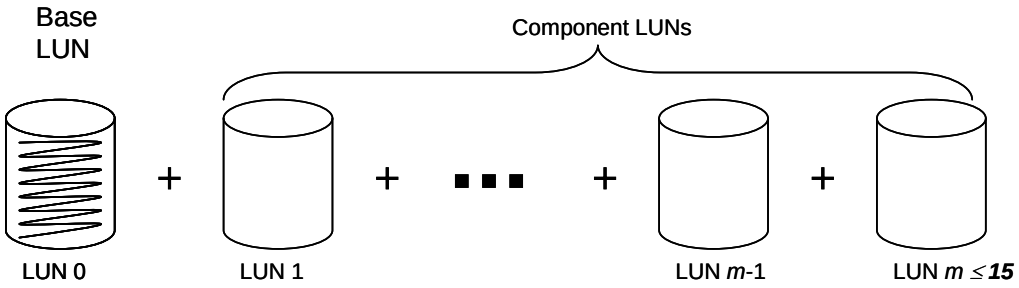


Figure 4 Concatentated MetaLUN: Base LUN and component LUNs

Figure 4 shows the relationship of the base LUN to the component LUNs in a concatenated MetaLUN. In a concatenated expansion, a maximum of 16 component LUNs (base LUN + 15 component LUNs) may be joined to a MetaLUN. Note that concatenated MetaLUNs typically have an m:m (where m ☐ 16) relationship between components and FLARE LUNs. That is, concatenation makes each component LUN into a MetaLUN component. Component LUNs may only be concatenated together one at a time.

Note there are CLARiiON-model and FLARE-revision dependencies on the maximum number of components that may be expanded. The figures above apply to CLARiiON CX4 with FLARE revision 28.0 and later. Consult your storage system's documentation for the attributes specific to MetaLUN component maximums for legacy CLARiiONs.

Maximum MetaLUNs

The number of MetaLUNs that may be created on a CLARiiON storage system is model-dependent. Entry-level models support fewer MetaLUNs than the Enterprise models. In addition, there is FLARE version dependency. Later revisions of FLARE support double the number of MetaLUNs for a particular model over earlier revisions.

Table 3 shows the maximum number of MetaLUNs per storage system for FLARE revisions supported on the CX4 model line.

Note that private LUN ID availability also affects the maximum number of MetaLUNs that may be created.

Table 3 MetaLUNs per CX4 storage system

FLARE REVISION	CX4-120	CX4-240	CX4-480	CX4-960
Maximum MetaLUNs FLARE 28.0 and later	512	512	1024	2048

The maximum number of MetaLUNs for legacy CLARiiONs with FLARE revision 26.0 and earlier is similar to that of FLARE 28.0 for the comparable model based on number of drives. For example, the maximum number of MetaLUNs for a legacy CX3-40F is 512. Review your storage system's documentation for the exact number.

Expansion process

When a MetaLUN is created, the base LUN and the separate component LUNs are joined into a new logical storage object. A MetaLUN storage object inherits the behavior and state of its base LUN. Inheritance is a way to form new instances of logical objects using previously defined objects. The base LUN is a FLARE LUN, in which case the MetaLUN has all the functions and features of a FLARE LUN. Among the things the MetaLUN inherits are the base LUN's identification both to the storage system and the host. It also inherits any in-use user data originally on the base LUN.

The LUN expansion process is transparent to the host. However, the time needed to complete an expansion can vary considerably. The time needed depends on the state of the base component, the type of expansion, and the in-use capacity of the base LUN. In addition, no more than four expansions can be scheduled to occur at the same time. Note that on busy RAID groups, performance may be adversely affected if four or more expansions are scheduled at the same time on the same RAID group.

Expansions of MetaLUNs used with active CLARiiON replication applications are not allowed. In general, you cannot start an expansion of a MetaLUN that is participating in either an active MirrorView or SAN Copy update. You must wait for these to complete. Review the online help in Navisphere® for specific recommendation on MetaLUN expansion and feature applications.

The following operations take place when a base LUN is expanded into a MetaLUN:

1. Create a MetaLUN storage object – All of the FLARE LUNs that make up the MetaLUN become private LUNs.
2. Set up single ownership – All the private LUNs are trespassed to the CLARiiON storage processor that owns the base LUN.
3. Set up the MetaLUN's attributes – The MetaLUN storage object inherits the host-facing attributes of the base LUN.

4. Renumber the private LUNs making up the MetaLUN – This is done for management of the array LUN ID (ALU) addressable space; the private LUNs making up the MetaLUN are renumbered, starting at the top of the storage-system-addressable LUN ID space and working backward.
5. Zero the unused capacity on the MetaLUN – If the expansion is a striped expansion, and the former base LUNs contain user data, it is restriped across the MetaLUN's components. The remaining new capacity is zeroed. If the expansion is a concatenated expansion, the newly appended component's capacity is zeroed.

The newly expanded MetaLUN storage object is the remaining addressable host object. The MetaLUN assumes the identity and attributes of the base LUN. The component LUNs cease to become separately addressable storage objects. The organization of the MetaLUN is affected by the type of expansion. If the expansion was striped, a single component is created. This component consolidates the base LUN, and all the FLARE LUNs participating in the expansion into a single component. If the expansion was concatenation, the MetaLUN will have as many components as there were component LUNs. (Concatenated expansion creates additional MetaLUN components.)

When the expansion is complete, you must perform a host O/S file system expansion to complete the operation and make all or some portion of the additional capacity usable.

Capacity

Each additional component added to the MetaLUN adds to the total available storage capacity of the MetaLUN. The capacity of a component is not a restriction in making it part of a MetaLUN. Component LUNs of any capacity may be joined together.

For example, in Figure 5 FLARE LUNs 0, 1, and 2 are each have a raw capacity of 1 TB. After LUN expansion, the MetaLUN LUN 0 would have a combined total raw capacity of 3 TB.

O/S dependency

Fully utilizing the capacity of MetaLUNs requires host O/S support. The host's addressing (32-bit or 64-bit) and the type of partitioning used are the most important factors.

A host partition is a LUN. It is a logical division of a physical drive or RAID group that functions as if it were a separate unit. After a partition is created, a file system is created in the partition. Note that some applications, such as databases, can use a LUN without a file system. Creating a file system on a partition is called formatting. Formatting can noticeably reduce a LUN's original raw capacity. It is possible to create MetaLUNs (and FLARE LUNs) with capacities that cannot be completely addressed by the host operating system. Be sure your host O/S can support the eventual capacity of the MetaLUN being created.

A 32-bit software architecture O/S, such as MS Windows Server 32-bit version, can address Master Boot Record (MBR) formatted partitions (LUNs) up to 4 TB in extent. A 64-bit software architecture, such as MS Windows Server 2008 R2 and later, can address GUID Partition Table (GPT) LUNs up to 16 exabytes in extent.

Linux and UNIX operating systems' addressing and file systems work in a similar fashion. The ext2/ext3 file systems can address 32 TB LUNs. A 64-bit file system implementation such as XFS or GPT must be enabled in the kernel for very large capacity LUN support.

Component element size

Matching the workload I/O size to the storage object is an important storage performance consideration. With FLARE LUNs this involves matching the LUN's underlying RAID group stripe size to the I/O. For example, if the I/O is 64 KB, a stripe size that is an even multiple of 64 KB (512 KB) is ideal. MetaLUNs also benefit from matching I/O to stripe size.

The component element size is the amount of data written to a component LUN in the MetaLUN component. The component element size is determined by the base LUN. With a striped MetaLUN the component element size specifies how much data is written to a component LUN before the next inline component LUN of the MetaLUN's stripe is addressed.

All MetaLUN components have a stripe width. The stripe width of a striped component is the sum of the RAID group stripe widths of the underlying component LUN RAID groups. The stripe width of a concatenated component is the same as its single component LUN. For example, consider a MetaLUN striped component made up of two component LUNs, where each LUN is bound to a five-drive RAID 5 (4+1) group. Each RAID group has a stripe width of five drives. This results in a stripe width of 10 drives for the component.

Striped expansion

The capacity of components is spread out as new addressable capacity across the MetaLUN's components using a data striping technique. In data striping, addressable data locations are sequentially assigned to component LUNs in a round-robin fashion. The addressable capacity of new component LUNs is merged into the base LUN's addressable capacity. This requires re-addressing the component LUN's data locations. Data is relocated from the base LUN component to the component LUNs containing relocated address locations. This results in the base LUN's in-use capacity being evenly redistributed across all the component LUNs in the first component.

Any capacity of the MetaLUN's component not in use after the base LUN's data has been restriped is overwritten with binary zeros. This "zeroing" is a data integrity and confidentiality feature. The zeros erase any previous data from the storage and set up for the component LUN's parity calculation. When zeroing is complete, parity and metadata are calculated for the LUN sectors. A detailed discussion of LUN data and

confidentiality features is found in the *EMC CLARiiON Storage System Fundamentals for Performance and Availability* document.

Additional capacity is not available immediately. If the base LUN has in-use capacity, that data must be "restriped" across the new component LUN. Restriping takes time. The interval is dependent on the in-use capacity of the base LUN, the number of component LUNs in the component, and the capacity of the original base LUN (or base component). The larger each one is, the longer the restriping takes. The additional capacity of the added LUNs is available to the host after the restriping process completes. The restriping process affects concurrent I/O, reducing throughput. However, the rate at which restriping takes place is low. It is also user-adjustable. It is prudent to plan carefully to ensure that the restriping process has time to complete during periods of low demand and *before* the additional capacity is needed.

Striped expansion requires *homogeneous* components. This means the base LUN and the component LUNs must all have the same LUN capacity. In addition, these LUNs must have the same underlying RAID group RAID level and drive type.

Throughput can be increased through striping. This is because either a single large or multiple small I/Os can be performed in *parallel* to the component LUNs of a striped MetaLUN. Parallelism involves more than one of the component's underlying RAID groups servicing a host's I/O request at the same time. This can result in a higher degree of parallelism and higher throughput.

Striping examples

The following examples show striped expansion with attention to data organization, capacity, and storage utilization.

Example 1. Striped expansion

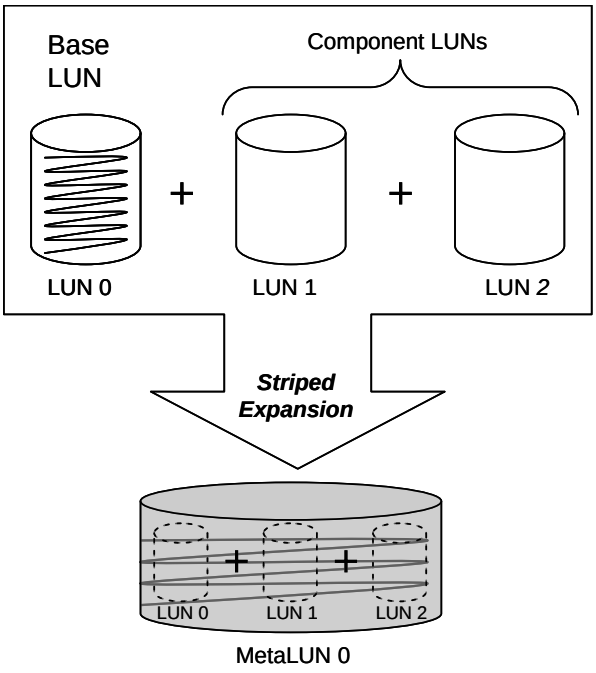
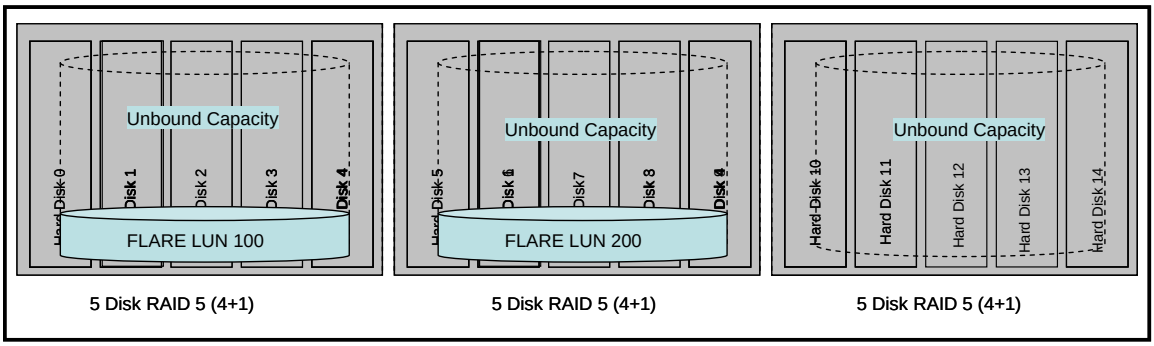


Figure 5 Example striped MetaLUN expansion

In Figure 5, the Base LUN (LUN 0) is expanded into a MetaLUN using Component LUNs 1 and 2. Assume all components are 150 GB FLARE LUNs and LUN 0 already contains 150 GB of user data. In the striped expansion, LUN 0's data is evenly redistributed by striping it across all the LUNs of the MetaLUN. The MetaLUN will have a total capacity of 450 GB (3*150 GB), of which 150 GB is in use. As a result of the striping, each FLARE LUN of the MetaLUN now contains 50 GB of the original user data coming from the base LUN. Any data that may have existed in LUN 1 and LUN 2 is overwritten and is destroyed. Also, note the MetaLUN is identified as LUN 0. LUN 0 is a single component MetaLUN. If 75 GB of additional data were written to MetaLUN 0, each component LUN would then have an additional 25 GB of data for a total of 75 GB of the overall 225 GB in-use capacity This capacity is not individually addressable. The MetaLUN as a whole, which is host-addressable, would have 225 GB in use of 450 GB total capacity.

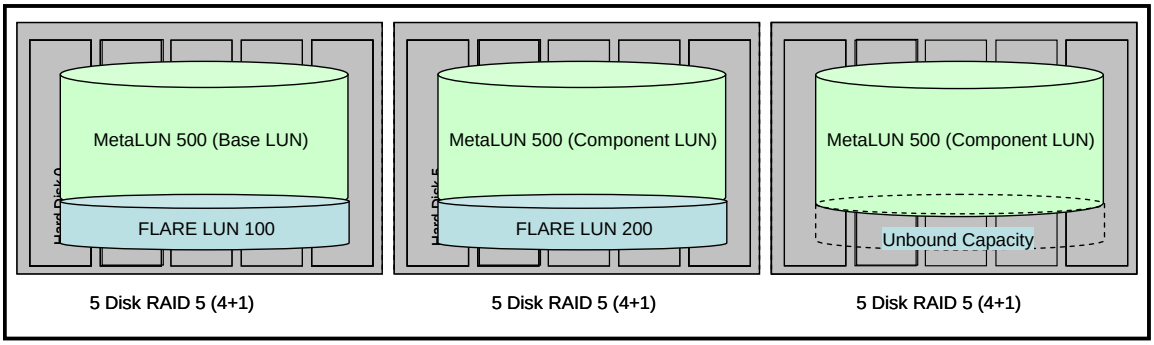
Example 2. RAID group utilization



15 Drive CLARiiON DAE provisioned with three 4+1 RAID groups

Figure 6 Capacity on identical RAID groups

Figure 6 shows a DAE with three RAID identical groups provisioned. Each RAID group has the same RAID level. Assume all the drives in the figure are the same type (EFD, Fibre Channel, SAS, or SATA). Assume FLARE LUNs 100 and 200 have the same capacity. All three RAID groups' unbound capacity is available for striping into a new single, large-capacity, high-performance MetaLUN.



15 Drive CLARiiON DAE provisioned with three 4+1 RAID groups

Figure 7 Homogenous stripped MetaLUN

Figure 7 shows one possible way to stripe the unutilized capacity of the three RAID groups into a single large-capacity, high-performance MetaLUN with LUN ID 500.

Note the above figure uses the example of a MetaLUN on a single DAE. MetaLUN component LUNs may be bound to *any* suitable RAID group with available capacity within the storage system. The single DAE example found in the figures is used purely for illustration purposes.

Striped expansion data organization

A MetaLUN stripe is not organized like a RAID group stripe. Understanding the difference can have a large effect on performance, particularly when component LUNs of a striped MetaLUN do not have the same underlying RAID group organization.

Striped expansion component stripe size

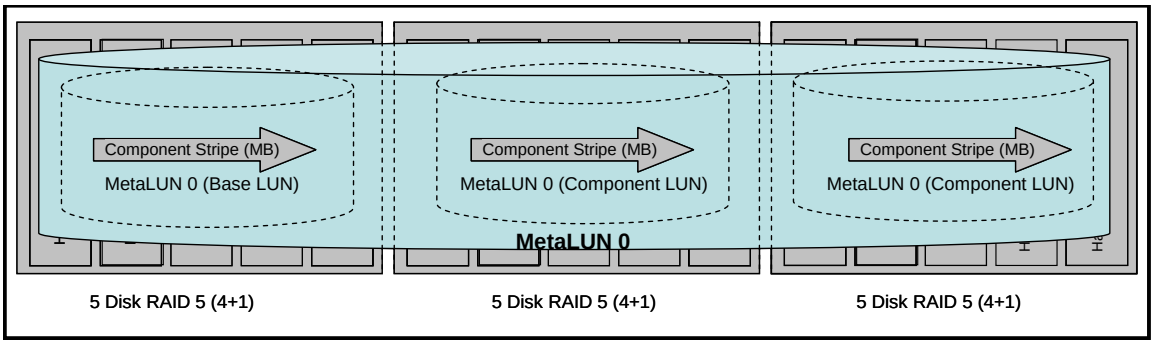
The component stripe size in a striped MetaLUN is determined by taking into account the MetaLUN's underlying RAID group organization and the MetaLUN stripe multiplier.

The stripe size of the first component LUN, the base LUN, sets the stripe size (using the MetaLUN stripe multiplier) for all component LUNs in that striped MetaLUN component. This applies to each MetaLUN component such that a MetaLUN is comprised of multiple striped components; each may have a different stripe size if their respective first LUN in each has a different organization. For example, in a RAID 5 group of five drives (4+1), the stripe size is 256 KB (64 KB for each disk in the RAID group – the parity drive is not counted). You then multiply the stripe size (256 KB) times the MetaLUN stripe multiplier (the default is 4) and you get the MetaLUN component stripe size, in this case 1 MB. This is the amount of data that is written to each LUN within that component, starting with the base LUN, before moving on to the

next LUN in the component. On each component LUN, parity is handled based on its RAID type.

Striped expansion restriping

With a MetaLUN stripe, data is striped from left to right across the drives of the LUN's underlying RAID group before moving to the next component LUN. The length of the component LUN stripe in bytes is the component stripe size of the base LUN times the MetaLUN stripe multiplier.



15 Drive CLARiiON DAE provisioned with three 4+1 RAID groups

Figure 8 Striped MetaLUN data organization

Note that the component stripe size is determined by multiplying the base LUN stripe size by the MetaLUN stripe multiplier. If there are a different number of drives in the underlying RAID groups of the striped MetaLUN's component LUNs, the restriping will not be ideally aligned. This misalignment between the component stripe size and the component LUN can have an adverse effect on performance.

Stripe Element Size Multiplier

You can use the *Element Size Multiplier* parameter to optimize I/O to the MetaLUN. The parameter describes the *data locality* of the I/O. Data locality is how widely the I/O address locations are distributed on a LUN. A high Element Size Multiplier means the I/O has a wide range of addresses. A low Element Size Multiplier means the I/O has a narrow range of addresses. The Element Size Multiplier is set at the component level. The effect of the Element Size Multiplier is to set the amount of data written or read from one component LUN in a MetaLUN component before advancing to the next component LUN. The amount of data written is denominated in stripe size units. The range is one to 255, with the default being four (4). The Element Size Multiplier is set when the MetaLUN is created. It cannot be changed afterward.

The calculation for the component element size is as follows:

Component Element Size = Element Size Multiplier * Base LUN Stripe Size.

For example, what is the component Element Size of a two component striped MetaLUNs whose base LUN is a four-drive RAID 5 group (4+1)?

- 4+1 RAID group Stripe Size = 256 KB (4 * 64 KB)

- Element Size Multiplier = 4 (Default)

  Component Element Size = 1 MB = (4 * 256 KB) * (1 MB / 1024 KB)

Stripe rules summary

The following rules summarize the MetaLUN component requirements for striped expansion:

- All LUNs in a component must be of the same user capacity.
- All LUNs in a component must be made up from RAID groups with the same RAID level.
- All LUNs in a component must be made up of the same underlying drive type.

Note that the most straightforward way to do a striped expansion is to start with FLARE LUNs that all have the same organization.

The components do not have to be identical. A striped expansion between two FLARE LUNs of the same capacity but from RAID groups with different numbers of drives is possible. This type of provisioning is *not* recommended if performance is one of the considerations for creating the MetaLUN. For example, assume there is a base LUN and a single component LUN, each bound to 250 GB. The base LUN may be bound from a RAID 5 group of five 450 GB Fibre Channel drives (4+1). The component LUN may be bound from a RAID 5 group of three 600 GB Fibre Channel drives (2+1).

Concatenated expansion

Concatenated expansion is the process of taking an existing base LUN and appending additional component LUNs to increase capacity.

Concatenation

The capacity of a concatenated component LUN is appended as new addressable capacity to the MetaLUN's base component. With concatenate expansion, in-use capacity residing on the base LUN remains in place. Additional capacity added in this fashion is available immediately. The addressable capacity of the new component LUNs is appended to the end of the base LUN's addressable capacity.

Concatenate expansion offers more flexibility of capacity expansion by allowing heterogeneous component expansion. This means the component LUN's capacity may be different. In addition, with certain restrictions, the RAID level underlying the component LUNs does not need to be the same. However, all the components must share the same drive type. For example all the drives in the RAID groups making up the components must be Fibre Channel. In addition, all the RAID groups making up the components must have the same level of protection.

Level of protection

Level of protection refers to the number of simultaneous drive failures a RAID group can sustain without data loss. Level of protection is not the same as RAID level. Having the same level of protection means the LUNs being concatenated must all be

unprotected, single, or double protected (see Table 2 for levels-of protection on page 3). Note from the table that RAID 6 LUNs can only be concatenated with other RAID 6 LUNs. Likewise, RAID 0 LUNs can only be concatenated with RAID 0 LUNs. However, RAID 5, RAID 3, and RAID 1/0 LUNs can be concatenated together because they have the same level of protection, despite being at different RAID levels.

Concatenation for high capacity utilization

It is not unusual for there to be RAID groups in a storage system with some unbound capacity. Concatenated expansion can be used to quickly maximize capacity utilization within the storage system. Unbound capacity may be unutilized for a number of reasons that may include odd capacity size, potential for I/O contention, drive type, provisioned RAID level, or location within the storage system.

When using concatenated expansion it is possible to quickly piece together non-contiguous storage space. This is done by binding LUNs using the unbound capacity in the RAID groups and using concatenated expansion to join them together. In addition, capacity created by concatenated expansion is immediately available for use; there is no delay to restripe data as in striped expansion.

There is no added performance with concatenated expansion, only additional capacity. Each component is a separately addressable space. Its performance is determined by the organization of its underlying RAID group organization.

Concatenation examples

The following examples show concatenated expansion with attention to data organization, capacity, and storage utilization.

Example 3. Data organization

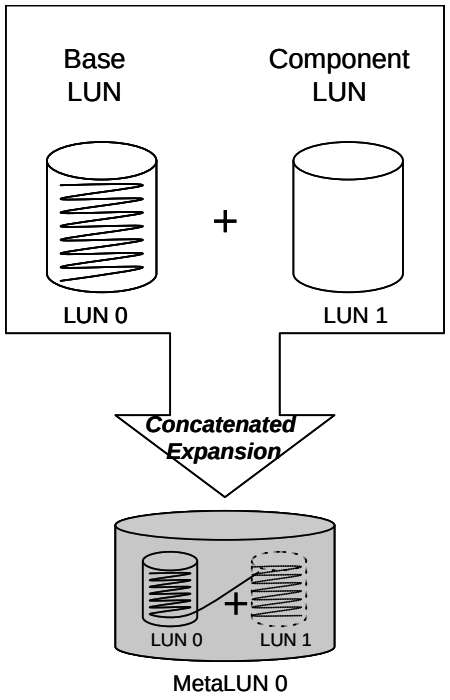


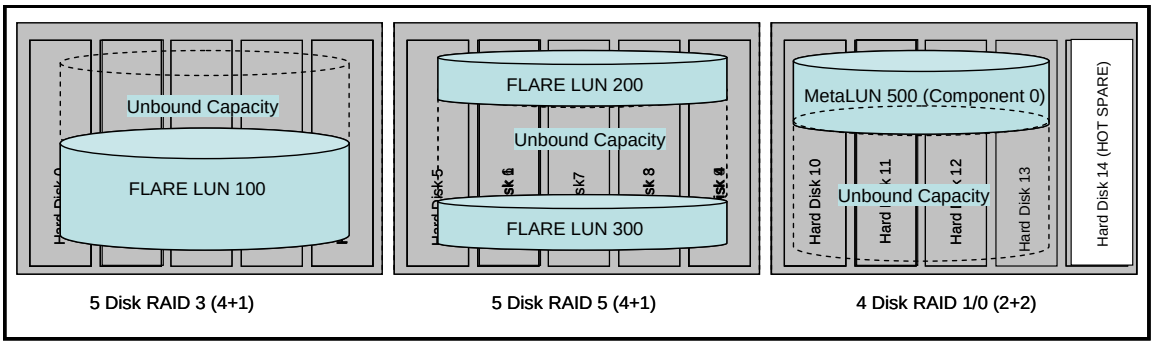
Figure 9 Example of concatenated MetaLUN expansion

In Figure 9 the Base LUN 0 is expanded into a MetaLUN using component LUN 1. Note that concatenated expansion takes place by appending one component LUN at a time.

Assume LUN 0 contains 150 GB of user data and is full. LUN 1 may be of any capacity; in this case we will assume it is 250 GB. In the concatenated expansion, LUN 0's data remains in the original base LUN. The MetaLUN will have a total capacity of 400 GB, of which 150 GB is utilized. The base LUN component of the MetaLUN contains the original 150 GB of user data; an additional 250 GB of unused capacity is available in the remaining component. Any data that may have existed in LUN 1 is overwritten and is destroyed. Also, note the MetaLUN is identified as LUN 0. MetaLUN 0 has two MetaLUN components.

If 75 GB of additional data is written to the MetaLUN, 50 GB of the data would be written to the original LUN 1 component filling it, and the remaining 25 GB would be written to the original LUN 1 component. The components would then have the following utilization: 150 GB and 25 GB. This capacity is not individually addressable. MetaLUN 0 would now have 225 GB of 400 GB utilized. The unused capacity would all be on the component that was originally LUN 1.

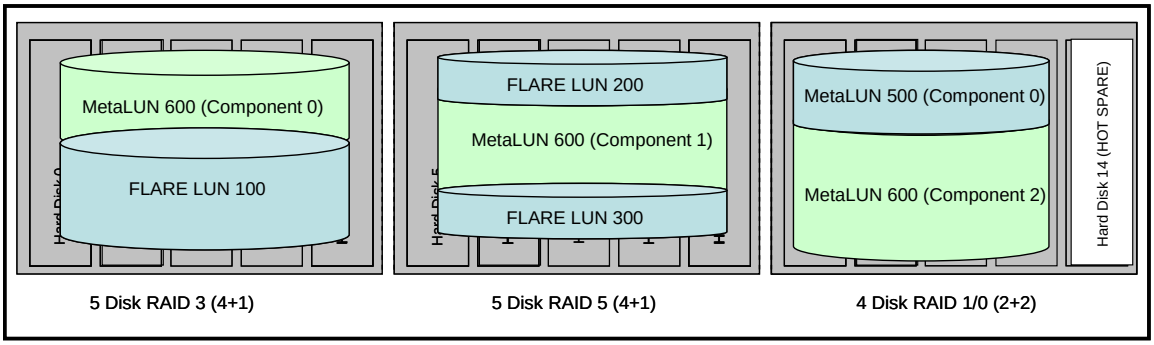
Example 4. RAID group utilization



15 Drive CLARiiON DAE provisioned with three RAID groups

Figure 10 Underutilized RAID group capacity

Figure 10 shows a DAE with three RAID groups provisioned. Each RAID group has a different RAID level. Assume all the drives in the figure are the same type. All three RAID groups have the same level of protection. Assume the unbound capacity of these RAID groups is different. All three RAID groups' unbound capacity is available for concatenation into a single, large capacity MetaLUN.



15 Drive CLARiiON DAE provisioned with three RAID groups

Figure 11 Hetrogenous concatenated MetaLUN

Figure 11 shows one possible way to concatenate the underutilized capacity into a single large capacity MetaLUN with LUN ID 600. Note MetaLUN 600's expansion would require two separate concatenations. The first would be concatenating the FLARE LUN that becomes Component 1 to the base LUN (LUN 600). The second is concatenating the FLARE LUN that becomes Component 2 to MetaLUN 600. In addition, the RAID group hosting Component 1 may benefit from a RAID group defragmentation before binding the FLARE LUN, which becomes Component 1.

Concatenation rules summary

The following rules summarize the component requirements for concatenated expansion:

- All LUNs in a component must be made up of underlying RAID groups with the same level of protection, not mixed.
- All LUNs in the concatenated MetaLUN must have the same underlying drive type.

Note the restriction on protection level. Consider a concatenated expansion between a base LUN of 1 TB and a component LUN of 333 GB. Both LUNs are FLARE LUNs. The base LUN may be bound from a RAID 5 group of five 2 TB SATA drives; this would be a 4+1. The component LUN may be bound from a six-drive RAID 1/0 (3+3) group of 500 GB SATA drives. Both components have a single-drive level of protection.

A concatenated expansion between a base LUN of 1 TB bound from a five-drive RAID 5 (4+1) group of 2 TB SATA drives, and a 333 GB component LUN bound to a six-drive RAID 6 (4+2) group of 500 GB SATA drives, is not allowed. This is because the base LUN is a RAID 5-based single drive protected RAID level. The component LUN is a double drive protected RAID level 6.

Since RAID level 6 is the only double protection RAID level, LUNs based on RAID level 6 are only expandable with other RAID level 6 components. RAID level 0 has the same restriction. It is the sole unprotected RAID level.

Exact expansion

The final capacity of either a striped or a concatenated expansion may be adjusted downward to meet specific capacity needs. This called an exact expansion. In an exact expansion, a MetaLUN is created that has a capacity less than the summed capacity of its component LUNs. An exact expansion leaves a portion of the MetaLUN total capacity unavailable and unaddressable. An exact expansion is more likely to be needed when a striped MetaLUN is expanded.

For example, remote mirroring with EMC MirrorView/ Asynchronous (MirrorView/A) requires matching 3 TB LUNs be on the local and remote storage systems. Capacity is available to bind two 1.6 GB FLARE LUNs on RAID 5 groups. If the two FLARE LUNs are expanded into a striped MetaLUN, it would have a natural capacity of about 3.2 TB. This would not match the 3 TB local storage system's LUN. Through and exact expansion, the MetaLUN's capacity can be reduced to 3.0 TB. This would leave 0.2 TB unavailable on the remote storage system's MetaLUN, but would maintain the symmetry of the mirror.

If the extra capacity is needed later, the extra capacity can be increased by using the **Properties** dialog box of the MetaLUN. There are two selections, **Maximum capacity** and **Other capacity**. From these you can increase the capacity of the MetaLUN from its current capacity.

Complex MetaLUNs

Complex MetaLUNs are used to create the very largest-capacity, highest-performing MetaLUNs. A complex MetaLUN is a MetaLUN whose component LUNs are joined by both expansion methods striping and concatenation. A complex MetaLUN consists of two or more MetaLUN components. These MetaLUN components are each made up of one or more component LUNs.

Components

Complex MetaLUNs have the same general organization of simple MetaLUNs created through a single expansion type. The base LUN (a component LUN) is always located in the base component. The base component is sometimes called *Component 0*. With complex MetaLUNs, the base component is always a previously created MetaLUN. If the MetaLUN was originally created by striped expansion, it has a single component, *Component 0*. If the MetaLUN was originally expanded by concatenation, it has at least two or more components. Figure 12 shows a MetaLUN that was expanded by concatenation, resulting in Component 0 and Component 1. In this example, the last component can be expanded through concatenation or striping using LUNs 1 though m, where m has a maximum value of 31.

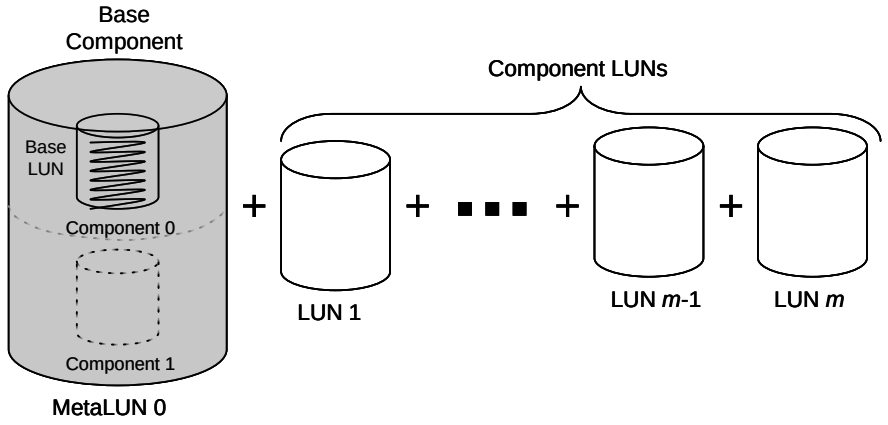


Figure 12 MetaLUN base component

Expansion

Additional FLARE LUNs can be either striped or concatenated onto the MetaLUN. This is done in the same way that FLARE LUNs are joined to a base LUN. FLARE LUNs may be striped onto the last component. However, no more than 32 component LUNs may make up any MetaLUN component. FLARE LUNs may be concatenated onto the last component. However, no more than 16 component LUNs may be concatenated as components in a single MetaLUN. Note that the restriction applies to components too. A CX4 MetaLUN can have no more than 16 components.

CLARiiON series and FLARE revision restrictions may apply to the maximum number of components allowed in a MetaLUN. Some legacy models (CX3 and earlier) support a maximum of eight components and each component can have a maximum of 16 drives. Review your storage system's documentation for specific values.

Component LUNs can be striped or concatenated onto the last MetaLUN components to create complex and high capacity MetaLUNs. The following two rules apply:

- Striping component LUNs onto a MetaLUN component increases the number of component LUNs per component. All striped additions to this component must be made before any concatenated expansion.

- Concatenating component LUNs onto a MetaLUN component increases the number of components per MetaLUN. Concatenation creates a new component for striped expansion. No additional striped expansions are allowed to previous MetaLUN components.

Large capacity MetaLUNs

Through a combination of striping and concatenating FLARE LUNs onto a MetaLUN, very large MetaLUNs can be constructed.

Component-wise expansion

Only individual FLARE LUNs may be used in expansion. Previously created MetaLUN components (with their component LUNs) cannot be detached and used in a MetaLUN expansion. In addition, all component LUNs are private LUNs. There are FLARE revision- and storage system model-dependent maximums for the number of LUNs per storage system (see the "Private LUNs" section). The availability of private LUN IDs may restrict the number of very large LUNs that can be created.

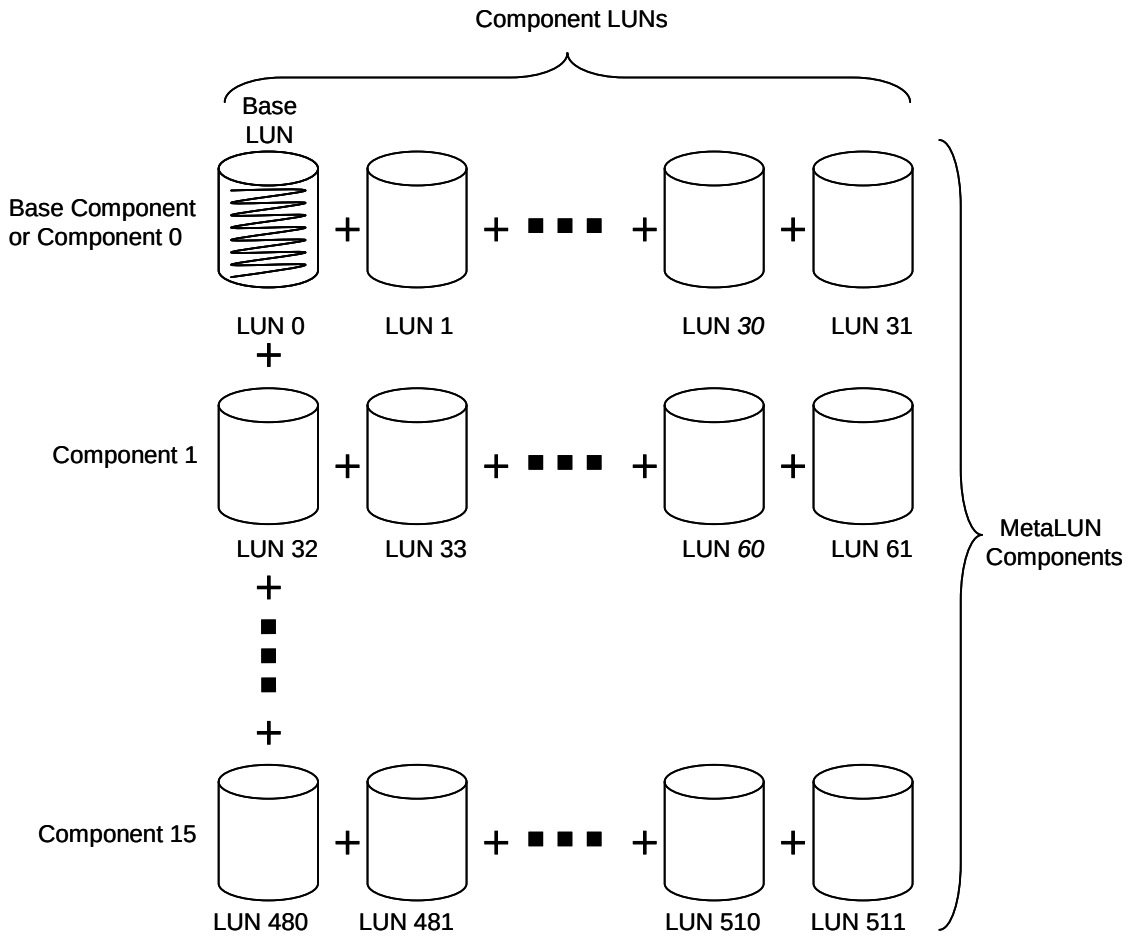


Figure 13 Largest MetaLUN: 512 component LUNs

Assuming the maximum number of dedicated component LUNs (512) using the entire capacity of the maximum number of drives in a RAID group (16), the largest MetaLUN that can be provisioned drive-wise contains 8,192 drives (16 drives * 512 component LUNs). This value exceeds the capacity of the largest CLARiiON storage system, the CX4-960, which can accommodate 955 drives (less the five vault drives).

Concatenating a FLARE LUN onto a striped component can change the capacity of components within the MetaLUN. In a striped component, all component LUNs must be the same capacity as the initial component LUN. In Component 0, the initial component LUN is the base LUN. In Components 1 thru 15 the initial component LUN may have a different capacity, because it is initially concatenated to its predecessor.

For example, in Figure 13 LUNs 0 through 31 in Component 0 (the base component) must have the same capacity. However, the initial component (LUN 32) of Component 1 can have a different capacity from the Base LUN (LUN 0). LUNs 33 through 61 must have the same capacity as LUN 32.

Data organization

A complex MetaLUN, with its combination of striped and concatenated components, has a unique data organization. The MetaLUN's address space is partitioned into tiers by the components. Each component can have different performance and availability characteristics. In any I/O it is first the underlying organization of the component LUN, and then the component, which determines the response time. In addition, the capacity of the MetaLUN is consumed component-by-component starting at the base component. For example, in a two-component MetaLUN, first the base component's capacity is fully consumed, then Component 1's capacity is used.

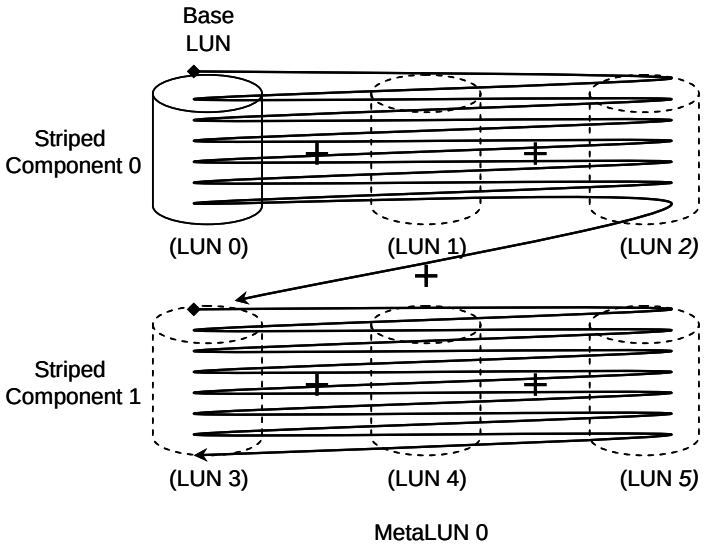


Figure 14 Data organization striped and concatentated MetaLUN

Figure 14 shows a two-component MetaLUN. Each component is made up of three component LUNs. Each component is striped. Capacity-wise, Component 1 augments the capacity of Component 0. Component 0 and Component 1 define a

single address space. With a workload of pure sequential writes, Component 0's capacity would be consumed first. Both components are striped components with all the performance benefits of striping across three component LUNs. Ideally, the component LUNs (LUN 0 through LUN 5) would share the same underlying RAID group organization on separate RAID groups. This would provide the most deterministic MetaLUN performance. However, this need not be the case. The RAID groups supporting the LUNs of Component 0 could be bound to five-drive RAID 5 (4+1) groups, while the RAID groups supporting the LUNs of Component 1 could be six-drive RAID 1/0 (3+3) groups. This organization would result in significantly different performance between I/O to the two components, which are a single address space.

Mixed expansion MetaLUN examples

The following section contains a multi-part example of the striped and concatenated expansion of a MetaLUN.

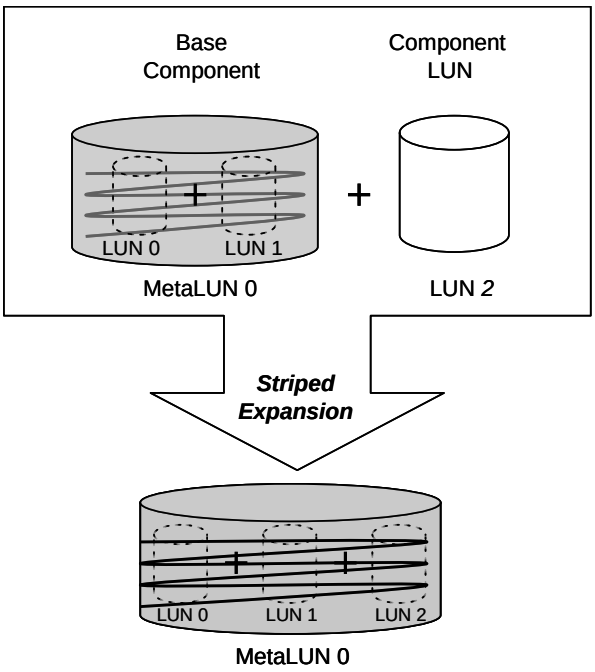


Figure 15 Striped expansion of striped MetaLUN

Consider the most common case, of adding a FLARE LUN to an existing striped MetaLUN, as shown in Figure 15. MetaLUN 0 is a single component MetaLUN originally made by striping together two FLARE LUNs (LUN 0 and LUN 1). MetaLUN 0 is expanded by striping a FLARE LUN (LUN 2) into it. LUN 2 has to have the same capacity, RAID level, and underlying drive type as the base LUN (LUN 0). In the expansion, the existing user data on MetaLUN 0 is restriped across the LUNs making up MetaLUN 0. MetaLUN 0 consists of a single component (Component 0) containing three component LUNs.

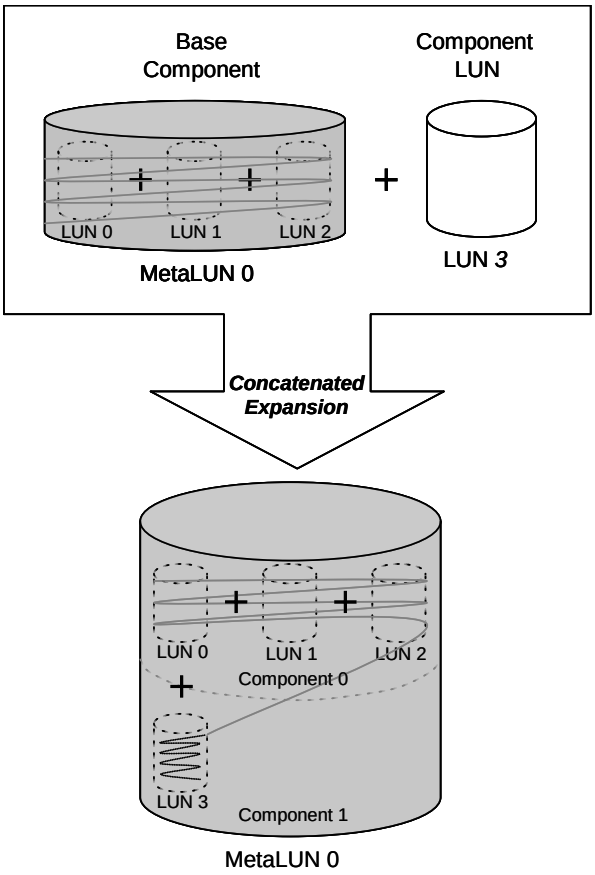


Figure 16 Concatenated expansion of a striped MetaLUN

Concatenated expansion of an existing striped MetaLUN is shown in Figure 16. MetaLUN 0 is a single component MetaLUN originally made up by striping together three FLARE LUNs (LUN 0, LUN 1, and LUN 2). This is the base component, or Component 0. MetaLUN 0 is expanded by concatenating FLARE LUN 3 onto it. LUN 3 must have the same level of protection as the base LUN (LUN 0). LUN 3 may have a different capacity from LUN 0. The newly concatenated LUN becomes Component 1. In the expansion, the existing user data on Component 0 is unaffected. If user data is sequentially written to MetaLUN 0, Component 0 would continue to be written. If Component 0 reaches full capacity, Component 1 would start to receive data.

Capacity-wise, Component 1 augments the capacity of Component 0. However, they do not share the same organization of their data. Component 0 is a striped component with all the performance benefits of striping across three component LUNs. Component 1 is a concatenated component. It has the performance of the original LUN 3. Note that LUN 3 may also have a different capacity from LUN 0.

Up to 14 (16-2) additional FLARE LUNs can be concatenated onto MetaLUN 0 one at a time.

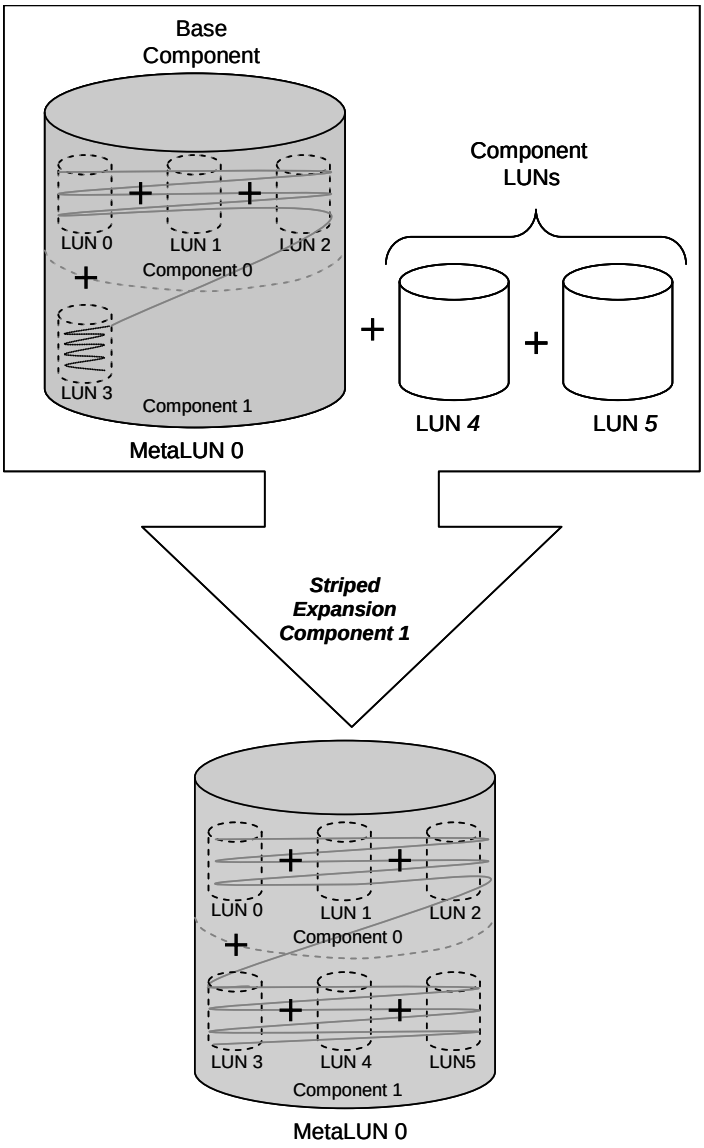


Figure 17 Striped expansion of a MetaLUN component

Figure 17 shows striped expansion of an existing concatenated MetaLUN. MetaLUN 0 is a two-component MetaLUN. Component 0 is striped and made up of three component LUNs. Component 1 is concatenated to Component 0 and is made up of a single component LUN. MetaLUN 0 is expanded by striping two FLARE LUNs (LUN 4 and LUN 5) into its Component 1. With the striped addition of the FLARE LUNs, Component 1 becomes a striped component. In the expansion, any existing user data on Component 1 is restriped across the three component LUNs now making up Component 1.

Capacity-wise, Component 1 augments the capacity of Component 0. However, they do not share the same organization of their data. Component 0 is a striped component with all the performance benefits of striping across three component LUNs (LUNs 0 through LUN 2). Component 1 is a separate component. It has the

performance of striping across a different set of three component LUNs (LUNs 3 through LUN 5). In addition, the capacity of Component 1 may be different from the capacity of Component 0.

Planning for MetaLUN expansion

A MetaLUN is a logical storage object designed by the user for capacity and performance. Its MetaLUN components and component LUNs are provisioned for this purpose. However, a MetaLUN may not be a static storage object. Both types of component have a relationship with each other in the types of drives that make up their RAID groups, the organization of the RAID groups, and the positioning of the component LUNs within the storage system that produce the needed capacity and optimal performance. Over the course of its useful lifetime the MetaLUN will likely be increased in capacity or be modified to meet workload performance requirements. This will mean adding MetaLUN components and component LUNs. The initial design of a MetaLUN needs to include provision for the MetaLUNs future expansion. This includes an understanding of:

- Under what circumstances will the MetaLUN need to be expanded?
- How often is the MetaLUN likely to be expanded?
- How much additional capacity (or performance) will likely be added in an expansion?
- Where will the additional LUNs that make up the expansion be located within the storage system?

For example, if a 20 TB MetaLUN needs to be increased in capacity by 5 TB, will an additional MetaLUN component be concatenated on, or will one or more existing MetaLUN components have additional component LUNs striped onto it? Will the additional capacity come from the storage system's reserve capacity, or will new DAEs with drives be purchased for the expansion? If unused capacity provides for the expansion, is there likely to be contention between the newly created component LUNs with existing FLARE LUNs or MetaLUN component LUNs?

Forward planning of a MetaLUN's future growth is an important part of its design requirement. It is prudent to overdesign, and underprovision, a MetaLUN. That is, lay out a MetaLUN design with either larger capacity or more performance than is needed, but only implement the required capacity and performance. The unused resources can be used as a template for the MetaLUN's future expansion.

MetaLUN capacity

Generally, the capacity of a MetaLUN is the sum of the capacity of its component LUNs. The maximum capacity of the component LUNs is determined by the user capacity of their underlying RAID groups.

You need to have a firm understanding of the MetaLUN storage object's lifecycle before planning to use any of the alternatives discussed in this section. This includes its initial capacity, and the future scaling of this requirement. Knowing where within the storage system a MetaLUN's capacity is allocated, and having plans from where future capacity shall be allocated from ensures the best capacity utilization and MetaLUN performance.

RAID group drive numbers and MetaLUNs

A component LUN's capacity is some fraction of the underlying RAID group's user capacity. User capacity (or data capacity) of the drives placed in a RAID group is less than the raw capacity of the drives due to the capacity consumed by the RAID group's data protection, data integrity mechanisms, and metadata.

A MetaLUN's maximum user capacity is determined by the maximum user capacity of the RAID groups of its component LUNs. A LUN can be some or all of the available capacity of its underlying RAID group. Note that a component LUN can always be bound with less than the maximum capacity of its underlying RAID group.

A lot of CLARiiON performance is determined at the RAID group level. Table 4 shows the recommended provisioning for the CLARiiON CX4 RAID groups. This recommended provisioning applies to the RAID groups of component LUNs in a MetaLUN. These RAID levels and number of drives provide an optimal balance of user capacity to raw capacity, bandwidth, and throughput performance over a broad range of I/O profiles, along with high availability.

Provision MetaLUN component LUNs from RAID groups containing modest numbers of drives. Do not use component LUNs from RAID groups with less than a total of four drives. Ideally, all component LUNs of a MetaLUN component should have the same number of drives and the same drive speed in their underlying RAID groups. Symmetrically provisioning RAID groups in a MetaLUN leads to the highest and most predictable performance.

Capacity utilization

Single FLARE LUNs have better capacity utilization than MetaLUNs. Be aware of the capacity being allocated to data protection within component LUNs. In some cases a single FLARE LUN will have a higher capacity utilization and the same level of protection as the same size MetaLUN. Note that the MetaLUN may have higher IOPS than the FLARE LUN. If performance is not a factor, and the storage requirement can be met with a single FLARE LUN, the RAID 5-based FLARE LUN is the most efficient use of storage.

For example, consider a dedicated FLARE LUN taking up the entire capacity of a single nine-drive level-5 RAID group. It has the same user capacity, but fewer drives than a MetaLUN made up of two component LUNs, each of which takes up the full capacity of a five-drive RAID 5 group. The FLARE LUN has a user capacity of eight drives using a total of nine drives. The MetaLUN has a user capacity of eight drives using a total of 10 drives.

Table 4 Component LUN RAID group provisioning

RAID level	Organization	User drives	Total drives
RAID 1/0	3+3	3	6
	4+4	4	8
RAID 5	4+1	4	5
	8+1	8	9
RAID 6	8+2	8	10
	10+2	10	12

The most commonly used RAID levels are 1/0, 5, and 6. Because the different RAID levels use different data protection schemes to provide levels-of-protection, the number of drives available for user data varies. The organization of RAID groups is shown with two numbers, separated by a plus (+) sign. The first number in the pair is the number of drive capacities available for data capacity (user drives). It is the capacity of these drives that make up LUN capacity. The second number is the number of drives dedicated to data redundancy. The sum of the two numbers in the pair is the total number of drives in the RAID group (total drives). For example, a 4+4 would be a RAID 1/0 group provisioned with eight drives, of which the capacity of four drives are available for user data.

Most file systems and database applications default to a 256 KB, 512 KB or 1 MB max I/O size. Ideally, a RAID group's stripe size should match or be an even multiple of the most common I/O sizes received from the host. The 4+4, 4+1, 8+1, and 8+2 RAID level organizations align this way naturally. These RAID group configurations are good candidates for creating optimally performing LUNs.

Component LUN provisioning

The user capacity of a RAID group is calculated by multiplying the number of user drives by the user capacity of the individual drives making up the RAID group.

For example, a 400 GB, 15,000 rpm Fibre Channel drive has a formatted data capacity of about 367 GB. A RAID level 1/0 (3+3) group would have a maximum user capacity of about 1101 GB (3 * 367). This approximate 1100 GB is the maximum capacity of a FLARE LUN bound on the RAID group. Striping two FLARE LUNs using the maximum user capacity of this RAID group's provisioning would create a MetaLUN with a maximum possible user capacity of 2.2 TB.

Component LUN capacity and MetaLUNs

A MetaLUN's capacity is the sum of the capacities of its component LUNs. However, the capacity of a MetaLUN cannot always be calculated simply by multiplying the capacity of the base LUN times the number of component LUNs.

MetaLUNs may be made up of one or more striped and concatenated components. A component is made up of component LUNs. In a striped component, all component LUNs must be the same capacity. In Component 0 the base LUN determines that

capacity. In Components 1 through 15 the first concatenated component LUN may have a different capacity from the base LUN or any of its predecessors.

Capacity example

For example, in Figure 18 assume the Base LUN (LUN 0) has a capacity of 500 GB. Because LUN 1 and LUN 2 are striped with LUN 0, they must also have a 500 GB capacity. Component 0 (not the entire MetaLUN) has a total capacity of 1.5 TB (3*500 GB).

Assume LUN 3 has a capacity of 1 TB. LUN 3 can have a different capacity from the Base LUN, because it was added to the MetaLUN by a concatenated expansion. MetaLUN 0 made up of Component 0 and Component 1, containing LUNs 0 through 3, has a capacity of 2.5 TB ((3 * 500 GB) + 1 TB).

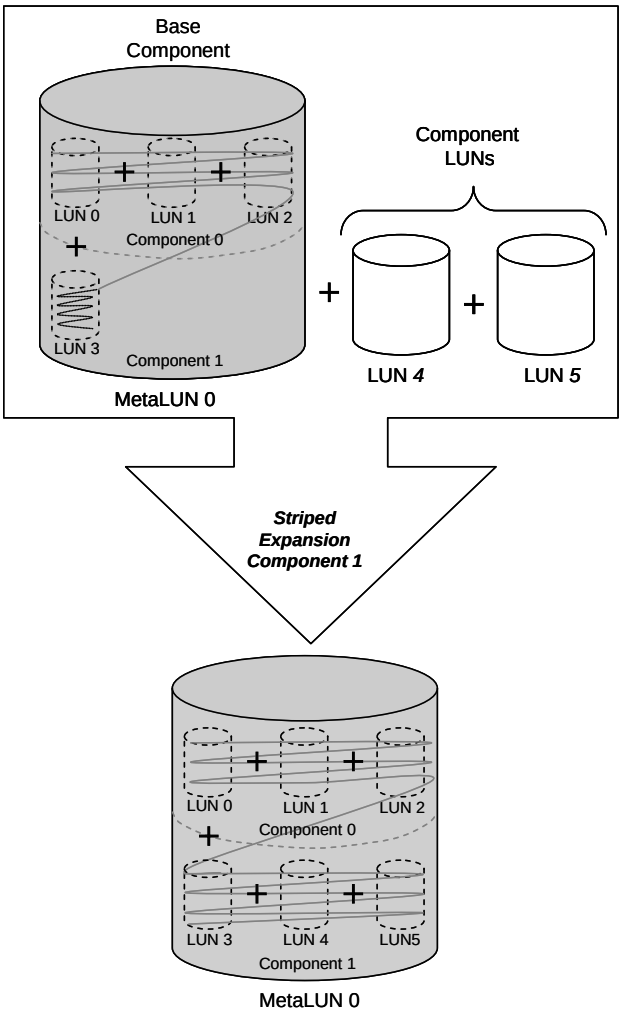


Figure 18 Striped expansion of Component 1 capacity example

A striped expansion of Component 1 with FLARE LUNs 4 and LUN 5 requires these LUNs have a 1 TB capacity. After the expansion, Component 1 will have a capacity of 3 TB (3 * 1 TB). MetaLUN 0 will have a total capacity of 4.5 TB ((3 * 500 GB) + (3 * 1TB)).

Note that while you can change the capacity of component LUNs across a MetaLUN's components, we do not recommend doing this. Ideally, the MetaLUN should have uniform performance with every I/O. Varying the capacities of component LUNs causes the performance to vary between the components. That causes the overall performance of the MetaLUN to vary depending on the component being written to or read from. To ensure consistent performance, keep the capacity of the component LUNs the same or as close as possible.

MetaLUN performance

The performance of a MetaLUN component is determined by the performance of its component FLARE LUNs. A FLARE LUN's performance is determined by its underlying RAID groups. So, MetaLUN performance is directly related to CLARiiON RAID group performance. You need an in-depth understanding of RAID group performance to get optimal MetaLUN performance.

In addition, both the MetaLUN expansion type and the location of component LUNs within the storage system affect performance, and the expansion type has a large effect on MetaLUN scalability. Also, you need to consider the relationship between the component LUNs within their RAID groups and the organization of the component LUN's underlying RAID group to achieve optimal performance.

Before considering any of the alternatives discussed in the following sections, you should specify the MetaLUN storage object's lifecycle. This includes initial performance, and the future scaling of this requirement. Knowing where within the storage system a MetaLUN's component LUNs are allocated, and having plans from where future component LUNs shall be allocated from, ensures the best MetaLUN performance and capacity utilization.

Generally, the recommendations found in this section apply to mechanical hard drives. MetaLUN performance with EFD-type drives is different. Consult the latest revision of *EMC CLARiiON Best Practices for Performance and Availability* for EDF performance recommendations.

Striped MetaLUNs for performance

Generally, the larger the number of the drives in a LUN's underlying RAID group, the greater the available throughput. The same is true for striped MetaLUNs; the more LUNs (which means more drives) in a stripped component, the better the performance. Of course this depends on a number of factors: RAID type; drive type and speed; organization of the RAID groups and LUNs; and the best practices inspired planning that went into the design.

Concatenated MetaLUN performance

Concatenated MetaLUNs do not have the performance benefits of striping; they are more capacity-oriented. The throughput of an unstriped concatenated component is that of a single FLARE LUN. Each separate component within a concatenated MetaLUN has a separate performance profile based on the RAID type, drive type, and speed. Depending on the configuration of the LUNs, the performance of a concatenated MetaLUN can vary significantly. However, for the most consistent MetaLUN performance, we recommend concatenating either the same or very similar LUNs together into MetaLUNs.

Match component LUNs with application I/O

The performance of a MetaLUN depends on the ability of its component FLARE LUNs to handle the I/O. An important consideration is to match the underlying RAID level of the component LUN's to the application's I/O workload. This requires knowing the workload's I/O:

- Type: Sequential or random
- Access: Reads versus writes
- Size: Large-block size or small-block size
- Threading model: Single versus multi-threaded

In addition, you need to understand the performance characteristics of the CLARiiON's RAID levels. Parity and mirror RAID levels handle type, access, size, and threading differently. For example, if the I/O workload sent to the MetaLUN is random, and 40 percent of the I/Os are 16 KB (small-block) writes, selecting component LUNs with an underlying eight-drive RAID level 1/0 (4+4) leads to better MetaLUN performance than five-drive, RAID 5 (4+1) component LUNs. Note this example assumes the 4+4 and 4+1 LUNs are of the same capacity. Also the performance benefit comes at the expense of additional drives per component LUN (see Table 4 on page 3).

EMC CLARiiON Best Practices for Performance and Availability contains guidance on provisioning FLARE LUNs from RAID groups that complement an application's I/O.

Component LUNs per MetaLUN

Optimal performance depends on the given application workload. An application's I/O affects the number of component LUNs needed in a MetaLUN to achieve optimal performance. The optimal number of component LUNs in the MetaLUN differs based on whether the I/O is primarily random or sequential.

The more drives there are in a LUN, the better the random I/O workload performance. We recommend that you distribute a MetaLUN's I/O across as many component LUNs as possible when the I/O is small-block random I/O. (Small-block is considered to be up to and including 16 KB.) These component LUNs need to have at least the recommended number of drives (see the RAID group drive numbers and MetaLUNs section). This engages many RAID groups in servicing the I/O. That is, within the

capacity requirement, provision MetaLUNs using the largest practical number of component LUNs. This may require provisioning a larger number of lower-capacity FLARE LUNs to make up the MetaLUN's capacity requirement. Four to eight component LUNs in a striped component MetaLUN should be tried first to get optimal performance. However, if there are not have enough concurrently outstanding I/O requests, fewer LUNs may be needed.

When the workload is primarily large-block (64 KB and greater) random or large-block sequential, the fewest practical number of component LUNs should be used to make up the MetaLUN. For large-block random I/Os or writes that bypass the write cache, the base LUN stripe size should also be matched to the I/O block size for best performance. Matching means the I/O block size should be the same or a multiple of the component LUN's underlying RAID group stripe size. To accomplish this matching, you must provision the underlying RAID groups with the correct number of drives.

Provision component LUNs for performance

MetaLUN performance is highly dependent on how the component LUNs are bound to their underlying RAID groups. All the performance best practices that apply to FLARE LUNs apply to MetaLUN component LUNs. This includes drive selection (type, capacity, and speed), number of drives, and back-end bus positioning. The EMC CLARiiON Best Practices for Performance and Availability white paper includes a detailed discussion of RAID group provisioning for performance.

Symmetrically provisioned component LUNs

For consistent and predictable performance, all the component LUNs should be symmetrically provisioned. This means they should have the same capacity and underlying RAID group organization.

Through concatenation and striped expansion it is possible to create MetaLUNs with component LUNs that have different capacities and RAID group organizations. Depending on the workload these differences can produce very different response times from the individual component LUNs. These differences can adversely affect the performance of the MetaLUN or other LUNs bound to the RAID groups.

For example, Figure 19 shows a striped MetaLUN with component LUNs having different RAID group organizations. MetaLUN 0 was created by striping together a 500 GB base LUN from a five 450 GB 15,000 rpm, Fibre Channel drive, RAID 5 (4+1) group with a 500 GB component LUN bound from a four 450 GB, 15,000 rpm, Fibre Channel drive, RAID 5 (3+1) group. Note the LUNs are the same capacity, the RAID level is the same, and the drives are the same type and speed. In addition, assume each component LUN receives an equal portion of the MetaLUN's I/O.

MetaLUN 0 is an allowed MetaLUN construction. However, the base LUN is supported by five drives (4+1) and the component LUN is supported by four drives (3+1). This organization leaves the component LUN with about 20 percent fewer available IOPS than the base LUN. This may result in a measureable difference in response times between I/Os going to addresses located on the MetaLUN's separate component

LUNs. See the section Stripe Element Size Multiplier for more information on data striping within a MetaLUN component.

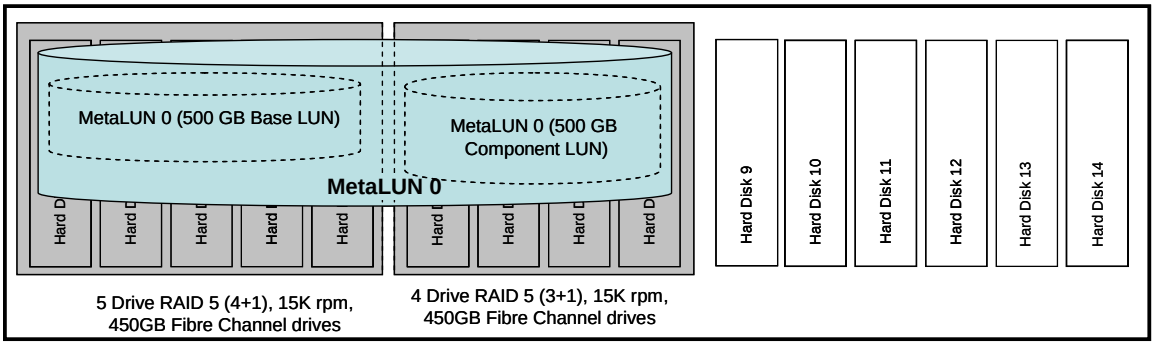


Figure 19 Example of an unsymmetrical striped MetaLUN

Symmetrically provisioned component LUNs and MetaLUN components give the same I/O performance across the entire address space of the MetaLUN. Try to make the component LUNs (which include the base LUN) identical, or as similar as is practical in:

- Capacity
- RAID group level
- Number of drives
- Drive speed

For example, as shown in Figure 20, if the MetaLUN's base LUN is 500 GB in capacity, bound to a five-drive, RAID 5 (4+1) group of 15,000 rpm Fibre Channel drives, try to ensure that all the MetaLUN's component LUNs are bound to RAID groups with the same organization.

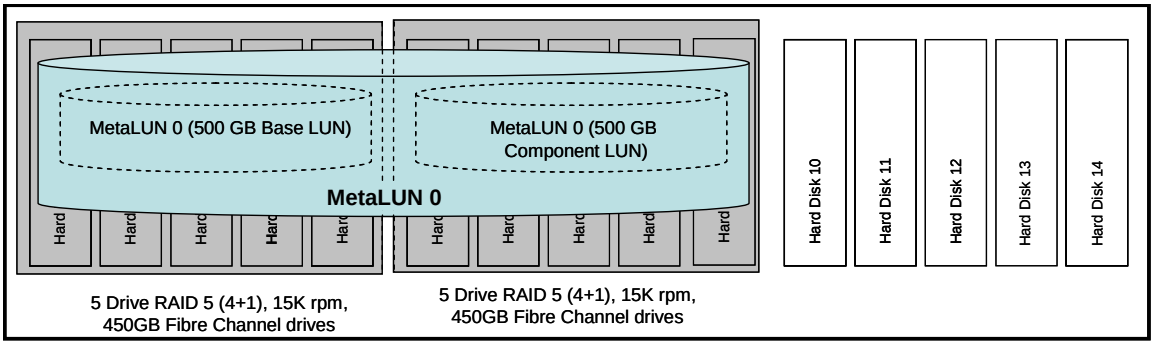


Figure 20 Example of a symmetrically provisioned MetaLUN

Symmetrically provisioned component LUNs are particularly important with striped MetaLUNs and within striped MetaLUN components.

Dedicated RAID groups

I/O contention occurs when the I/O of different LUNs bound to the same RAID group interferes with each other in competition for the RAID group's resources. For optimal MetaLUN performance, it is important to avoid linked contention and drive contention (see the Partitioning RAID groups with LUNs section). The highest performance is achieved by selecting component LUNs made up from dedicated RAID groups. That is one component LUN per RAID group. One component LUN per RAID group avoids any potential for either linked or disk contention.

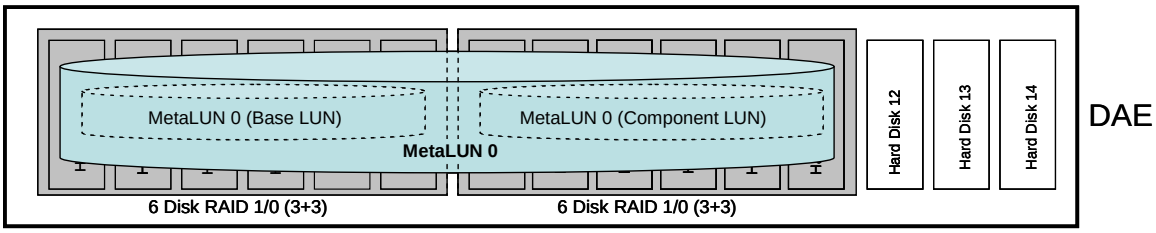


Figure 21 High performance: One component LUN per RAID group

Partitioned RAID groups

It is not always practical for component LUNs to have dedicated RAID groups. It is important to plan carefully to ensure the best performance when component LUNs are bound on RAID groups with other LUNs.

If component LUNs are bound to partitioned RAID groups, ensure the MetaLUN's component LUNs are bound to separate RAID groups. No component LUN should share a RAID group with a peer component LUN from the same MetaLUN.

Be aware of the workload sent to LUNs sharing the MetaLUN's RAID groups. Position MetaLUN component LUNs and separate FLARE LUNs with complementary I/O workloads together. An example is all LUNs bound to the RAID group performing small-block, random I/O. For sequential workloads, it is even better to avoid a large number of LUNs on any given RAID group.

When creating or expanding a MetaLUN, all component LUNs are trespassed to the storage processor owning the base LUN. When creating MetaLUNs with large numbers of component LUNs owned by different storage processors, additional planning may be required. This may be needed to get an even distribution of load on the storage system's back-end buses. When a RAID group gets partitioned between storage processors, consider the underlying read and write loads separately and then together on all LUNs on the partitioned RAID group. The best solution is to balance expected I/O across the storage system (not just the single storage processor).

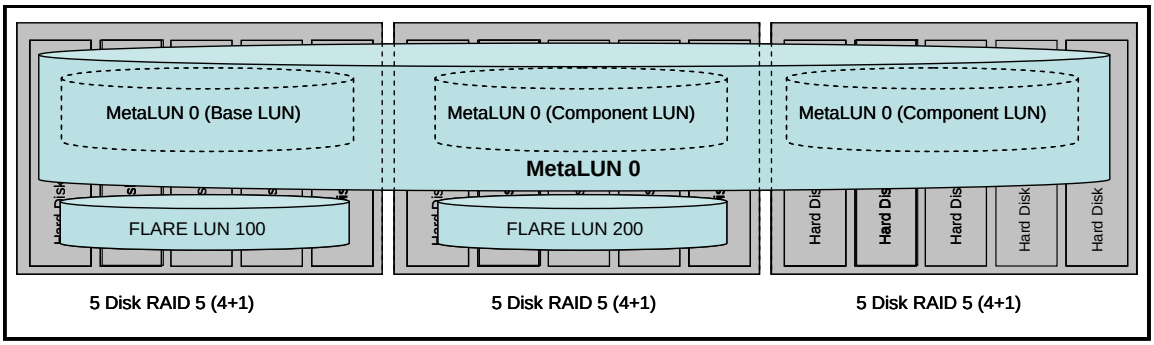


Figure 22 MetaLUN with distributed component FLARE LUNs

Figure 22 shows an example of a MetaLUN with distributed component LUNs. MetaLUN 0 is a striped MetaLUN made up of three components LUNs. Each of MetaLUN 0's component LUNs is provisioned on a separate five-disk RAID group. With this distribution, the MetaLUN's performance benefits from the maximum number of drives.

In this example, MetaLUN 0's base LUN and one of its component LUNs share a RAID group with separate FLARE LUNs. The storage processor owning MetaLUN 0 may not own either FLARE LUN 100 or 200. To avoid unpredictable performance, MetaLUN 0 and these FLARE LUNs should have complementary I/O workloads. For the best and most consistent performance, EMC recommends that you not provision non-MetaLUN LUNs in RAID groups used for MetaLUNs. Otherwise, ideally FLARE LUN 100 and FLARE LUN 200 should be inactive when MetaLUN 0 is servicing its workload to avoid any contention. However, if MetaLUN 0, FLARE LUN 100, and FLARE LUN 200 are all active at the same time, it would be advantageous if they were servicing applications performing a complementary small-block, random I/O workload.

RAID group defragmenting

LUN fragmentation occurs when gaps of unused capacity exist between LUNs in a partitioned RAID group. These gaps increase the seek distance between LUNs within the RAID group. Increased seek distance adversely effects response time. In addition, gaps leave less contiguous space in the RAID group for new LUNs or for expanding existing LUNs.

Through Navisphere, a RAID group can be defragmented to compress the gaps making the entire RAID group's in-use capacity contiguous. This brings the LUNs bound to the RAID group closer together, reducing seek times. It also allows for the creation of new, and possibly larger, LUNs within the recovered capacity.

Note that RAID group defragmentation is not file system defragmentation. (They are frequently confused.) RAID group defragmentation has no effect on the positioning of application data within a LUN. In addition, RAID 6 groups cannot be defragmented.

To get the best performance and the highest capacity utilization out of the storage system's RAID groups, verify how fragmented a RAID group may be before binding a FLARE LUN for use as a MetaLUN component.

The error of the stacked MetaLUN

The *stacked MetaLUN* (sometimes called the *vertically striped MetaLUN*) is a common MetaLUN provisioning *error*. A stacked MetaLUN is a MetaLUN where two or more of its component LUNs are striped together on the same RAID group. Any sequential I/O in a stacked MetaLUN causes linked contention. This has a severe adverse effect on the MetaLUN's performance.

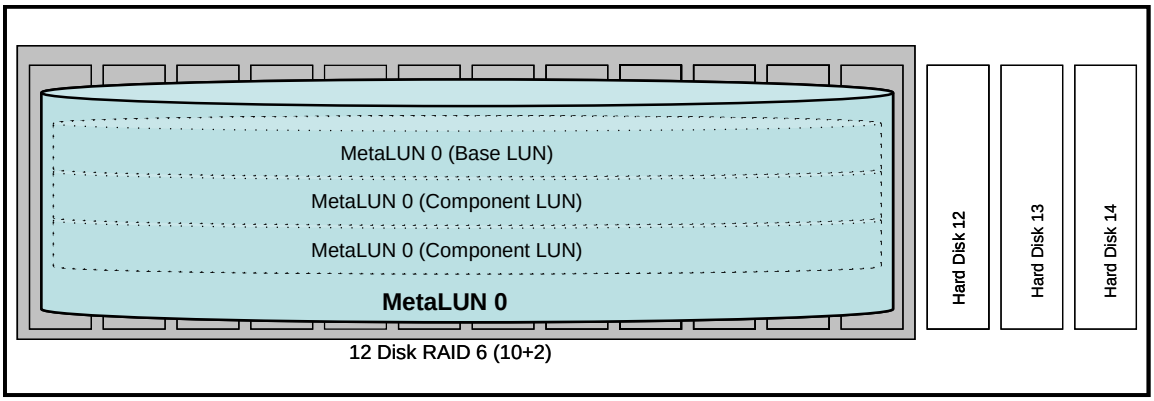


Figure 23 Stacked MetaLUN: A common MetaLUN provisioning error

Figure 23 shows an example of a stacked MetaLUN. MetaLUN 0 is a single component striped MetaLUN. All of MetaLUN 0's three component LUNs (which includes the base LUN) are located in Component 0. The MetaLUN's component LUNs are all bound on a single RAID group. Any widely distributed sequential I/O sent to the component LUNs sharing a RAID group will degrade the performance of all component LUNs. Widely distributed means I/O not restricted to the addresses a single component LUN.

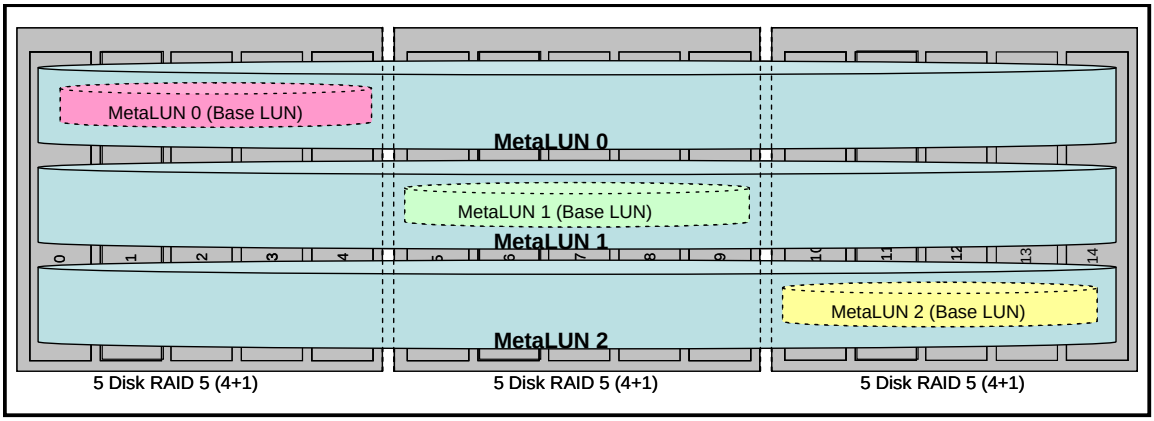
You cannot migrate a private LUN that is in use by a MetaLUN. To correct this, the MetaLUN's base LUN must be migrated like a normal LUN. This requires creating a LUN (or MetaLUN) the same size or larger than the existing MetaLUN.

Note a stacked MetaLUN only has poor performance for striped component LUNs. For stacked concatenated component, LUNs performance is not greatly affected.

Staggered base LUNs

When more than one MetaLUN is provisioned on a storage system, EMC recommends that you segregate the MetaLUNs to their own RAID groups. (RAID groups should be provisioned just with MetaLUNs.) When provisioning the component LUNs in the same RAID group always bind one LUN from each MetaLUN per RAID group – in the order that they were created – and do not use fewer LUNs than the number of RAID groups that you are using.

When creating more than one MetaLUN from a set of RAID groups, bind the base LUN of each MetaLUN onto a different RAID group. This distribution more evenly spreads the I/O load of the MetaLUN across the RAID groups when LUN I/O rates are higher for addresses in the first MetaLUN stripe. For example, file system metadata is located on the base LUN in the first physical megabyte of address range. The positioning of the non-base LUN component LUNs is not important.



15 Drive CLARiiON DAE provisioned with three 4+1 RAID groups

Figure 24 Staggered MetaLUN base LUNs

For example, in Figure 24 three MetaLUNs (0 thru 2) are created from component LUNs using three 4+1 RAID groups in the same DAE. Provision the base LUN of the first MetaLUN (MetaLUN 0) to be the first RAID group. The base LUN of MetaLUN 1 gets bound on the second RAID group. The base LUN of MetaLUN 2 is on the last RAID group of the set of RAID groups.

Component LUNs on vault drives

The first five drives 0 through 4 in DAE0 in a CLARiiON CX4 series array are the system drives that contain the saved write cache in the event of a failure, the storage system's operating system files, the Persistent Storage Manager (PSM), and the FLARE configuration database. The first four drives in the AX series are reserved for this purpose. These drives are also referred to as the system drives or the vault drives.

When possible, avoid binding LUNs with heavy workloads onto the vault drives. Their usage by applications may affect overall system performance. When the vault drives are used for component LUNs, ensure that they are not subject to high bandwidth or high IOPs workloads. Review *EMC CLARiiON Best Practices for Performance and Availability* for specific recommendations on vault drive usage.

MetaLUN cache settings

For MetaLUNs, you should use the CLARiiON read and write cache setting techniques used for high-performance FLARE LUNs.

Cache settings are made per LUN. Configure the read and write cache for component LUNs for performance before using them as part of a MetaLUN. Generally, enable both read and write cache. Do not overallocate your read cache at the expense of the write cache. Less than or equal to 20 percent of cache memory allocated to read cache works well for the majority of workloads. However, there are exceptions. Very large systems with primarily large-block, sequential workloads can benefit with read cache allocations up to 30 percent of cache memory.

Set the cache page size to the size of the predominant I/O size. If the I/O size has a lot of variability, use the default I/O size.

Also, ensure that the write cache's write-aside value is the default 2048. However, be aware of exceptions. For example, EFD-type drive-based LUNs have cache disabled by default. Another exception is that performance for some I/O profiles may benefit from changes like increasing the cache page size. See the *EMC CLARiiON Best Practices for Performance and Availability* for the latest recommendations on LUN cache configuration for performance.

Stripe expansion rate

Stripe expansion takes time. It also consumes a certain amount of the storage system's resources. The MetaLUN's data is always available during the expansion. Plan ahead before starting an expansion because the MetaLUN's performance will be somewhat lower during its execution.

During the expansion, the data on the original MetaLUN is restriped across the component LUNs of the new, larger capacity MetaLUN. The time required to complete striping is dependent on the:

- Storage system's background workload
- Priority: Low, Medium, High, or ASAP (As Soon As Possible)
- Original capacity of the MetaLUN or base component

There are four priorities (Low, Medium, High, and ASAP) for restriping. The Low, Medium, and High priorities economically use the system's resources, but have the longest duration. The ASAP priority accomplishes the operation in the shortest period of time. However, it also causes a higher load to be placed on the drives, and additional storage processor CPU load. This may adversely affect host I/O performance to those drives.

The default expansion rate priority for a MetaLUN under load is High. If there is no I/O workload on the MetaLUN or the RAID groups containing the expansion component(s), the expansion rate priority is automatically reset to ASAP. This means that more than one rate may be in use during the duration of the expansion. For example, the default High rate is used during hours while the MetaLUN is in use, and the ASAP rate after when the MetaLUN is detected to be idle.

The provisioning of the original base LUN or base component is the next most important factor. Performance is dependent on how fast you can read from and write to the original component LUNs at the same time, since the new expanded

component LUN is only being written to. In the homogenous striped expansion, sources and destinations likely have the same organization. This makes the capacity of the base LUN or component the driving factor. Table 5 shows the expansion rates for the different priorities.

Table 5 Striped MetaLUN striping rates for FLARE revision 30

PRIORITY	DRIVE RATE (MB/S PER DRIVE)
Low	0.10
Medium	0.14
High	0.26
ASAP	1.30

In the table, all values are per drive. Rates are only for mechanical drives. (EFD-type drives are not shown.) The rate is not affected by drive type (Fibre Channel, SAS, or SATA). RAID level does not affect the rate, except for ASAP. In addition, per drive means the total capacity of the RAID groups, including both data and data protection consumed by the LUN. For example, a 1 GB LUN on a five-drive RAID 5 (4+1) group consumes about 1.25 GB (1 GB data + 0.250 GB for parity data) of its RAID group's overall capacity.

Example of striped MetaLUN expansion duration estimate

This example shows how to estimate the duration of a LUN expansion.

How long is required to complete the striped expansion at Medium priority of a 150 GB MetaLUN with an additional 50 GB component LUN for a total 200 GB capacity MetaLUN?

The original MetaLUN is made up of three component LUNs each of 50 GB. Each component LUN resides on a five-drive RAID 5 (4+1) group. The total in-use RAID group capacity of each component LUN of the MetaLUN is 62.5 GB (50 GB user data + 12.5 GB parity data).

- Original LUN Data Capacity: 150 GB
- LUNs Data Protection Capacity: 37.5 GB (3 component LUNs * 12.5 GB parity data)
- Expansion Rate (Medium): 0.14 MB/s (from Table 5)
- Original Number of Drives: 15 (3 component LUNs * 5-drive (4+1) RAID group)

Time = (Original LUN Data Capacity + LUN Data Protection Capacity) / (Expansion Rate * Original Number of Drives)

24.8 hrs. = (((150 GB + 37.5 GB) * 1024 MB/GB) / (0.14 MB/s per drive * 15 drives)) / 3600 sec/hr

MetaLUNs and alignment

I/Os that cannot be accommodated by a single RAID group stripe element cross to a second disk. These are called *disk crossing* I/Os. It is normal for I/O larger than the stripe element size to cross drives. In the case of I/O smaller than the stripe element size, disk crossing I/Os may have a modest adverse effect on performance.

Alignment is how the data is laid out on the set of a RAID group's stripe elements. It ensures that I/O more neatly fits into stripe elements.

When creating a MetaLUN for user data, only the base component needs stripe alignment. If using the LUN alignment offset, the MetaLUN alignment offset is applied when the MetaLUN is created. When expanding a MetaLUN, no other adjustment is needed. When using a host's file system alignment tools, the adjustment should be made when the user partitions the LUN for the first time.

If a MetaLUN is being used as a target for the CLARiiON replication applications SnapView or MirrorView, leave the MetaLUN alignment offset to zero. That is, do not align the base component using the storage system-based alignment.

EMC Navisphere Analyzer

Metrics are important in evaluating a MetaLUN's performance. MetaLUN performance data is available through Navisphere Analyzer for FLARE revisions earlier than 29.0. Metrics available for MetaLUNs are a superset of FLARE LUN metrics.

Two additional metrics are available for MetaLUNs:

- LUN read crossings per second
- LUN write crossings per second

LUN read and write crossings per second are important for understanding how data is being laid out across the FLARE LUNs in a MetaLUN component. Like disk crossings, the objective is to keep LUN crossings as low as possible for best performance.

Analyzer gives the user two views: LUN view and RAID group view. Using these views, users can understand the performance behavior of the FLARE LUNs that make up a MetaLUN and relate that to the physical spindles on which they reside. The LUN view shows the MetaLUN and all of the objects underneath it. The RAID group view shows to which RAID group each LUN belongs.

Knowing which RAID group each FLARE LUN belongs to is important in understanding the performance of MetaLUNs. It also allows users to see the locality of their data within the MetaLUN and its associated activity. Locality is how closely the I/O addresses are grouped together on the RAID group's drives. Additionally, users now have the option to launch Analyzer charts directly from a RAID group and storage group in the Navisphere Manager tree.

MetaLUN performance data is taken from a combination of the Storage Processor Agent and the MetaLUN driver. All data collected from the MetaLUN driver is current at the time of the request, while data collected from the agent contains the performance data from the last agent poll. The default real-time polling rate is 60 seconds, while

the default Analyzer archive polling rate is 120 seconds (120 in FLARE 24.0 and later, 600 in earlier versions).

Users should be sure their specified Storage Processor Agent polling rate of FLARE, which is managed through the Management Server Setup Page, is always less than or equal to the Analyzer archive polling rate. Failure to do so will result in inconsistencies in the recorded data. If the Analyzer archive polling rate is less than the SP Agent polling rate, the Analyzer provider will record performance data from the agent for multiple periods before the agent performs its poll of FLARE to refresh the data. The time axis displayed in Performance Detailed charts notes that time that the Analyzer provider collected the data. The following metrics are available for MetaLUNs:

- Utilization
- Queue Length
- Response Time
- Total Bandwidth
- Read Bandwidth
- Write Bandwidth
- Total Throughput
- Read Throughput
- Write Throughput
- Read Size
- Write Size
- Average Busy Queue Length
- Service Time
- FLU Read Crossings
- FLU Write Crossing

The remainder of the metrics are calculated by a combination of information coming from the MetaLUN driver and the Storage Processor Agent or purely from the SP Agent. The metrics listed previously will not be affected by the relative polling rates of Analyzer and the SP Agent, whereas the remainder of the metrics could be affected if the Analyzer polling rate is less than the SP Agent polling rate of FLARE.

Information on how to use Navisphere Analyzer can be found in the Navisphere Manager online help and the *EMC Navisphere Analyzer Administrator's Guide* available on Powerlink.

MetaLUN availability

Like MetaLUN performance, its expansion type, and the component LUNs bound, affect a MetaLUN's availability. The availability of a MetaLUN is the availability of its component LUNs. Plan ahead. Provisioning decisions made at the LUN level determine the MetaLUN's overall availability.

A MetaLUN has greater availability than a FLARE LUN with the same number of drives in the single failure scenario. This is because a MetaLUN is made up of multiple component LUNs; any single failure is "compartmentalized" to the one LUN of many contributing to the MetaLUN's performance. However, in the unlikely event of an entire RAID group failing, for example in the case of a double drive failure in a RAID 5 group, the entire contents of the MetaLUN would become unavailable. Depending on a number of factors, a failure(s) will have a greater or lesser effect on performance. Hardware failures are the primary availability concern. The adverse effects of failures are mitigated by:

- Drive provisioning
- RAID group level and provisioning
- Proactive hot sparing
- Rebuilds

In addition, failures that are not related to drives need be considered. Examples are network failures, and hardware failures that affect only one of the storage system's storage processors.

Understand that before considering any of the alternatives in this section, a firm understanding of the MetaLUN's lifecycle is needed. This includes initial availability needs, and the effect of performance and capacity scaling on the requirement for availability.

RAID group provisioning

On the CLARiiON a large portion of availability is at the RAID group level. The relationship between the MetaLUN's component LUN's RAID groups, and the organization of those RAID groups need to be considered to achieve the highest availability.

Drives

The selection of the drives that make up the RAID groups of the MetaLUN's component LUNs has a large effect on performance and availability. Generally, use EFD, Fibre Channel, and SAS drives for mission-critical and high transaction rate workloads. EFDs, whose memory-based storage has no moving parts, have the highest availability. Use SATA drives for large-block sequential, less-critical applications with lower IOPS workloads.

RAID groups

The selection of a RAID group's RAID level also affects availability. The higher the level of protection in a components RAID groups, the higher the availability. To summarize in order of availability from highest level of protection to lowest:

1. RAID 6
2. RAID 1/0
3. RAID 5, RAID 3, RAID 1
4. RAID 0

For example, a MetaLUN made up exclusively of LUNs bound to RAID 6 groups has a higher availability than a MetaLUN made up of LUNs bound to RAID 5 groups. RAID 6 has a level of protection of 2. RAID 5 has a level of protection of 1. A LUN based on RAID 6 groups can sustain two simultaneous failures. The RAID 5 LUN can sustain only one.

RAID 0 offers no data protection. You should not use this RAID level for LUNs containing any data of value.

In addition, the number of drives in the component RAID group affects availability. Generally, RAID groups with fewer drives have higher availability. This maxim is based upon the statistics of the mean time between failure (MTBF) of hard drives. However, the number of drives in a RAID group needs to be balanced with capacity requirements, particularly the amount of raw capacity of a RAID group dedicated to data protection. Data protection is provided by either mirroring or parity. Following the RAID group drive provisioning listed in Table 4 on page 3 provides a good compromise between capacity, availability, and performance.

Drive and RAID group summary

Following the recommendations above, you can see the highest availability can be achieved with a 10 EFD (8+2) RAID 6 group. This RAID group has the most reliable drives, in the highest level of protection RAID level group. However, this type of extreme availability provisioning may not be practical, and would only be advisable in the most extreme cases.

Practical choices include balancing the higher availability of Fibre Channel or SAS drives with the single drive level of protection of RAID 5 by provisioning with five Fibre Channel or SAS drive (4+1) RAID groups. Another practical example would be to trade off the lower availability of SATA drives with the double drive level of protection of RAID 6 by provisioning with 10 SATA drive (8+2) RAID groups for primarily sequential workloads. These trade-offs must also be tempered by their performance implications.

Provision the RAID groups that make up your MetaLUN's component LUNs to account for the availability you believe is needed, as well as for capacity and performance.

Proactive hot sparing

Hot spares are drives used to replace failing drives in any protected RAID group with bound LUNs. Provision enough hot spares to protect your MetaLUN's data. Maintain at least one hot spare for each drive type on the storage system. Drive types are EFD, Fibre Channel, SAS, and SATA. In addition, each type can have a different capacity and rotational speed (rpm). Hot sparing is not a CLARiiON requirement for normal operation. However, it is a prudent choice if you wish to increase overall storage system availability.

Rebuilds

A single drive failure in any of the RAID groups within the MetaLUN degrades the performance of I/O requests addressed to the component LUN(s) that are bound to the RAID group with the failed drive. All the other components continue as before until the MetaLUN is rebuilt. Striped MetaLUN component performance is likely to be affected more than concatenated components. In addition, depending on the level of protection, while a MetaLUN is operating with a failed drive, there is a chance of data loss should there be an additional failure before the initial failure is corrected.

A rebuild replaces the data of the failed drive onto the newly operational replacement drive. If there were no hot spares configured of appropriate type and size, no rebuild occurs. With an available hot spare, a rebuild is a two-step process: rebuild and equalize. During the rebuild step, all LUNs on the RAID group with the failed drive are rebuilt. The LUN's data is reconstructed using the mirrored or the parity data from other drives in the RAID group. The RAID group remains in a degraded state until the failed drive is replaced; and then the failed drive's RAID equalizes from the hot spare or rebuilds from the other remaining drives in the RAID group plus parity or its mirror drive, depending on the RAID level.

Because a MetaLUN contains several LUNs, each with its own underlying RAID group, under certain conditions multiple separate rebuilds could occur at the same time. In addition, it is possible for a single RAID group to have more than one rebuild executing at the same time. This is possible on RAID 1/0 and RAID 6 groups.

A rebuild is a prioritized operation. The available priorities are ASAP, High (default), Medium, and Low. Priority is set at the LUN level. Rebuild times depend on a number of factors, including the rebuild priority, presence and location of an appropriate hot spare, drive type, drive size, workload, and RAID group type. During the equalize step, which occurs after the faulty drive is replaced, the hot spare is copied to the replacement drive. More than one rebuild at the same time can have an adverse effect on MetaLUN performance.

During a rebuild, a MetaLUN's operation will be degraded. It is hard to predict how much it will be degraded. However, a multi-RAID group striped MetaLUN component is not affected as severely by a rebuild as a FLARE LUN. The MetaLUN's larger number of available IOPS can absorb the overhead of the rebuild operation in a production environment more easily. If the rebuild setting is the default High the rebuilding RAID group will itself incur less than 10 percent degradation of the overall MetaLUN. That

is, it will run very close to normal. In case of a concatenated MetaLUN component, where locality shifts over time, the effect of a rebuild might be the same as on a FLARE LUN.

For an ASAP rebuild, the throughput decrease in a MetaLUN is higher. It can be roughly calculated as a percentage by dividing the number of drives in the RAID group with the failed drive, by the number of drives in the MetaLUN component with the failed drive. For example, assume a MetaLUN made up of a single component, containing two five-drive RAID 5 groups, with a single failed drive. The component totals 10 drives. The failed RAID group has five drives. The approximate percentage of throughput lost in an ASAP rebuild would be 50 percent (5 drives / 10 drives * 100).

Failure trespasses

When a MetaLUN is created, the ownership of all the component LUNs is trespassed to the storage processor owning the base LUN. The component LUNs become private LUNs of the storage processor.

A single storage processor always owns all the component LUNs of a MetaLUN. Should a failure trespass occur, the MetaLUN and all its underlying component LUNs are trespassed to the peer storage processor. Trespassing a MetaLUN can shift a large number of component LUNs with their ongoing I/O to the peer storage processor in a single operation. This may increase the peer storage processor's utilization.

Performance headroom

When planning the provisioning of a MetaLUN to meet its workload, a margin of the MetaLUN's I/O capacity should be held in reserve under normal operating conditions for handling:

- Hardware and software failures
- Brief, large, unanticipated increases in the workload
- Maintenance activities

This margin is typically called performance headroom. It is overbuilding in case of emergencies. In particular, there should be enough spare IOPS left over in the MetaLUN while it is executing its workload to handle at least one RAID group rebuild without reducing I/O performance below acceptable levels. This is particularly important when a MetaLUN component has less than four component LUNs, or less than about nine drives in its stripe width.

Performance headroom should be extended to the storage processor level, too. It is always prudent to have the storage system's I/O roughly balanced between both storage processors. However, as a result of some failure events resulting in a trespass, a single storage processor assumes a portion of the I/O workload of its peer. Under rare circumstances, it is possible for a peer storage processor's workload to double. Having built-in performance headroom at the storage processor level

softens the performance degradation of a single storage processor assuming a larger portion of the storage system's the I/O workload(s).

MetaLUN alternatives

Typically, a MetaLUN's creation results from the need to expand the capacity of an existing FLARE LUN. A MetaLUN may also be used for performance reasons, where not enough bandwidth or IOPS is available from an existing FLARE LUN. A MetaLUN may be used when a very large capacity LUN is specified. The CLARiiON provides a lot of flexibility in its features providing capacity and performance. There are alternatives to MetaLUNs. Likewise, there are host-based capacity and performance features to consider, storage virtualization solutions being among them. Before committing to a MetaLUN provisioning strategy, it may be helpful to consider the options provided by:

- Host-based Logical Volume Managers
- FLARE LUNs
- Virtual Provisioning

A Logical Volume Manager (sometimes abbreviated as LVM or VM, but not to be confused with a Virtual Machine (VM)) is a host-based storage virtualization software application.

A FLARE LUN can be created from a single RAID group and expanded through LUN migration.

Virtual Provisioning maps address spaces into the capacity provided by a pool of drives. This presents an application with the virtual capacity of the pool.

Before considering any of these alternatives, you should have a firm understanding of the storage object's lifecycle. This includes initial capacity and performance, and the future scaling of the requirements.

Logical Volume Managers

Volume Managers abstract drives into virtual "volumes" for use by the host. LVMs create pools of drives into Volume Groups (VGs). VGs are partitioned into logical volumes (LVs) which are virtual drives presented to the host application. The LVM pool is managed at the host level, and not the storage system. The host's administrator has a lot of flexibility in portioning out pool capacity to host applications and users. In addition, most LVMs provide many advanced management features, such as replication. CLARiiON LUNs are provided to LVMs as VGs where they contribute their capacity, performance, and data integrity features to the pool.

Note that LVMs can be used with MetaLUNs, where the MetaLUN provides the VG to the LVM.

Volume Managers are supported by all host O/Ss. For example, Microsoft Windows Server's LVM is Logical Disk Manager, and Linux is supported by the Open Software

Foundation (OSF) LVM and Enterprise Volume Manager. Windows, Linux and UNIX are also supported by Symantec Veritas Volume Manager (VxVM).

The capacity of a LVM is limited by the host operating system's implementation. Consult the LVM's user documentation for specific amounts.

The performance of an LVM vs. a MetaLUN is a highly qualified comparison. The workload and the number of drives dedicated to the storage object have important effects. In addition, there are many ways to configure both the LVM and the MetaLUN. The following comparison statements assume the same number of mechanical hard drives up to about 16.

- For small-block random I/O, a MetaLUN and a LVM have similar performance for both reads and writes.
- For large-block sequential I/O, an LVM has better performance than a MetaLUN. This is particularly true for multi-threaded I/O when the VGs are spread across the storage system's two storage processors.

As the number of threads and drives increase, an LVM is likely to have a performance advantage over a MetaLUN. This is because:

- The LVM's I/O can be spread across the resources of both the storage system's storage processors. (The MetaLUN is restricted to one.)
- The LVM can scale across the entire storage system.
- LVMs can span across more than one storage system for an even higher scalability.

Be aware LVMs operate at the expense of host CPU resources. Also, management of the pool is the responsibility of the host's LVM administrator. This includes VG provisioning that abets the needed performance.

FLARE LUNs

LUNs are a fundamental component of the CLARiiON architecture. A FLARE LUN provides a straightforward solution to most storage requirements. FLARE LUNs have the highest performance of any storage object. They can have large capacity, and do not require additional planning or provisioning to set up, or additional labor to maintain. A single RAID group defines a FLARE LUN's maximum capacity and performance. The largest RAID group can have 16 drives. However, there are a large number of drive types available, each with different capacity and performance points. For example, very high performance can be achieved by creating LUNs based on EFD-provisioned RAID groups. High capacity can be achieved by creating LUNs based on large capacity (1 TB) SATA drive provisioned RAID groups.

The capacity of the largest FLARE LUN is that of the largest, highest-capacity drive dedicated RAID group. The maximum size of a LUN is dependent on the maximum number of drives in CLARiiON RAID group - 16. Secondly, it is dependent on the RAID level chosen. RAID 5 has the highest ratio of user space to raw capacity. Finally,

there is drive size. At this writing, the 2 TB SATA is the largest-capacity CLARiiON drive. That number of drives, RAID level, and drive capacity produce about a 30 TB (raw) LUN. Note that file system formatting will noticeably reduce the raw capacity to user (file system) capacity.

However, availability should be considered. It may not be prudent to provision a 16-drive, RAID 5 (15+1) LUN. For example, a higher availability 28 TB (raw) capacity LUN can be created using a 16-drive RAID 6 (14+2) with 2 TB SATA drives.

The performance of a large FLARE LUN vs. a MetaLUN is again a highly qualified comparison. The workload, and the number of drives dedicated to the storage object have important effects. In addition, there are many possible ways to configure both the FLARE LUN and the MetaLUN. The following comparison statements assume the same number of mechanical hard drives, where the drive count is less than 16.

- For small-block random I/O, a MetaLUN and a large FLARE LUN have similar performance for both reads and writes.
- For large-block sequential I/O, a FLARE LUN has similar performance to a MetaLUN with sequential reads. A MetaLUN has better performance than a FLARE LUN for sequential writes. This is particularly true for multi-threaded I/O where the MetaLUNs component LUNs are spread across different RAID groups.

As the number of threads increases, a MetaLUNs have a performance advantage. This is because the I/O can be spread across multiple RAID groups on the MetaLUN. Also, the largest FLARE LUN can only be bound to a 16-drive RAID group. A MetaLUN is needed when the performance or capacity requirement exceeds that number of drives.

However, high IOPS can be achieved using FLARE LUNs bound to EFD-based RAID groups. A single EFD has about 10 times the IOPS of a mechanical hard drive. An EFD LUN can substitute for a MetaLUN with a much larger number of hard drives in some performance situations.

Virtual Provisioning

Virtual Provisioning provides virtualization of LUNs at the storage system level. A virtually provisioned LUN is called a thin LUN (TLU). Thin LUNs present more capacity to an application than is physically available. The presentation of capacity not physically available avoids overprovisioning the hosts and underutilizing the storage system's capacity. When a thin LUN eventually requires additional capacity, its capacity is nondisruptively and automatically added from a reserve storage pool that has previously been provisioned with drives. Provisioning allocates drives to the pool. In addition, the storage pool's capacity can be nondisruptively and incrementally added to with no effect on the pool's thin LUNs.

Virtual Provisioning is an optional, licensed feature on the CLARiiON. That is, it must be purchased separately with the CLARiiON.

Thin LUNs have different capacity restrictions than FLARE LUNs. The minimum capacity of a thin LUN is one block (512 bytes). The maximum capacity of a thin LUN is 14 TB.

A MetaLUN offers better performance than a thin LUN under all threading models and numbers of drives. The performance of a MetaLUN is also more predictable than a thin LUN's. This is because a thin LUN is sharing the pool's capacity and performance with other workloads. However, thin LUNs scale easily. Their usage requires little planning and maintenance. For workloads with modest performance requirements, they represent an easy-to-use alternative to MetaLUNs.

MetaLUN management

The controls for managing MetaLUNs are integrated into the Navisphere Manager's (FLARE rev. 29 and earlier) User Interfaces (UIs) and the Secure Command Line Interface (naviseccli). Expand operations are conducted in the UI through an Expand Storage Wizard, and the navseccli.

Host-facing and trespass attributes are managed at the MetaLUN level, just as they are for FLARE LUNs. The following attributes can be set at the MetaLUN level in the same manner they are set for FLARE LUNs. They can be set at the time the MetaLUN is created or can be modified later (except where noted), using the Properties dialog box in the UI or the `metalun–modify` command in naviseccli.

- MetaLUN name (later modification)
- Default/current SP owner(later modification)
- Auto-assign (later modification)
- Alignment offset (not changeable)

There are other MetaLUN-specific attributes. All of these are specified at the time of MetaLUN creation, and a subset of them can be changed at a later time through the **Properties** dialog box in the UI or by using the MetaLUN –modify command. The MetaLUN-specific attributes are:

- Element size multiplier
- Expansion rate (changeable)
- Preserve data
- User capacity (changeable)

Finally, there are attributes that are still controlled at the FLARE LUN level. These values can be set independently for each FLARE LUN within a MetaLUN:

- Cache settings:
  - Enable/disable
  - Prefetch parameters

    - Write aside I/O
    - Cache retention
- Rebuild priority
- Verify priority
- LUN Name

Details on the exact sequence of provisioning commands can be found in the *Navisphere Manager Version 6.XX* online help for FLARE revisions 29.0 and earlier. To learn about CLI commands refer to the *Navisphere Command Line Interface (CLI) Reference*. All documents are available on EMC Powerlink.

EMC Navisphere User Interface (UI)

The Expand Storage Wizard and MetaLUN Properties dialog boxes contain all of the controls for MetaLUNs within the UI. Note that the same options are available for both FLARE LUNs and MetaLUNs; however, the terminology for the way they are dismantled is different. FLARE LUNs continue to have the Unbind option, while MetaLUNs have a Destroy option. Choosing the Expand option launches the Expand Storage Wizard.

Expand Storage Wizard

Whether the device is a FLARE LUN or a MetaLUN, the operation to expand a LUN is the same. When a FLARE LUN is expanded, its object type changes to a MetaLUN, as discussed in the Simple MetaLUNs section. Nevertheless, the user steps required to add additional capacity are identical for both FLARE LUNs and MetaLUNs. The Expand Storage Wizard walks the user through six steps:

1. Introduction
2. Select method of expansion (radio buttons)
    - Stripe
    - Concatenate
3. LUN selection
4. Capacity selection
    - Current capacity
    - Maximum capacity
    - Other
5. Attribute specification (initial expansion only)
    - MetaLUN name
    - SP owner
    - Expand priority

- Element Size Multiplier
- Alignment offset
- Auto assign

6. Summary

When managing through the UI Wizard, the UI will present options that are within the configuration rules.

MetaLUN properties

The UI provides a general storage object properties dialog box.

The General tab of the dialog box provides users with all of the general attributes of the MetaLUN. From this tab, users can set attributes such as:

- MetaLUN name
- Default (SP) owner
- Enable auto assign
- Expansion priority
- User capacity

You can increase the user capacity in as many operations as needed, until the maximum capacity is reached. Once at maximum capacity, additional FLARE LUNs must be added to increase the total capacity before the user capacity can be increased again. User capacity cannot be decreased at any time.

EMC Navisphere Command Line Interface (CLI)

The root command is `metalun` to control MetaLUN operations. There are several flags for expansion, destruction, modification, and information gathering.

```
metalun
  -expand
  -destroy
  -modify
  -info
  -list
```

Refer to the *Navisphere Command Line Interface (CLI) Version 6.X Reference* manual on Powerlink for the complete set of MetaLUN flags and options. Following is an example of the MetaLUN –info command that gives an overview of the MetaLUN driver information.

```
C:\Program Files\EMC\Navisphere CLI>java -jar navicli.jar -address
128.221.12.13
-user admin -password admin -scope 0 metalun -info
Can a LUN or a MetaLUN be expanded on this system:  Yes
```

```
Number of MetaLUNs in the system:  1

Maximum Number of MetaLUNs per system:  256

LUNs that are participated in MetaLUNs:  1022, 1023

LUNs that are available for expansion:  1, 2, 3, 4, 5, 6, 7, 8, 9, 11,
12, 13, 1

4

Maximum Number of components per MetaLUN:  8

Maximum Number of LUNs per component:  32
```

Navisphere CLI getlun and chglun commands

Traditional Navisphere CLI commands getlun and chglun will execute against MetaLUNs. There are fields and options within each of the commands that are not valid for MetaLUNs. In these cases, the storage system will return a constant—0—or N/A in fields that are not relevant for MetaLUNs.

Navicli getlun

Statistics returned by the getlun command that are relevant to MetaLUNs are:

- UID
- LUN ID
- LUN name
- State
- Host parameters:
  - Auto-trespass
  - Auto-assign
  - Default/current SP owner
  - Capacity

Navicli getlun –ismetalun

An additional option has been added to the legacy getlun command: -ismetalun. This option allows users to decipher LUN types in the storage system (FLARE LUNs versus MetaLUNs). To preserve existing scripts and applications, the output from the –ismetalun option is not included in the return strings of the getlun command. The –ismetalun option must be explicitly called out. Following is an example of this option with its output.

```
C:\>navicli -h 128.221.17.50 getlun -ismetalun

LOGICAL UNIT NUMBER 15

Is Meta LUN:      NO


LOGICAL UNIT NUMBER 2044
```

```
Is Meta LUN:      NO
LOGICAL UNIT NUMBER 2045
Is Meta LUN:      NO
LOGICAL UNIT NUMBER 2046
Is Meta LUN:      NO
LOGICAL UNIT NUMBER 2047
Is Meta LUN:      NO
LOGICAL UNIT NUMBER 9
Is Meta LUN:      YES
LOGICAL UNIT NUMBER 80
Is Meta LUN:      YES
```

There are five FLARE LUNs and two MetaLUNs in this storage system. It is likely that LUNs 2044 – 2047 are the FLARE LUNs that comprise MetaLUNs 9 and 80. To find out the exact structure of each MetaLUN, users would use the MetaLUN –list command.

Navicli chglun

Three values can be manipulated within MetaLUNs with the chglun command:

- Auto assign
- Default SP owner
- LUN name

The remainder of the values pertain to cache settings and rebuild priority. These values can be manipulated for MetaLUNs by directing the chglun command to each of the FLARE LUNs that make up the MetaLUN. Attempts to change these parameters on a MetaLUN will fail silently.

MetaLUN shrink

LUN shrink is the ability to reduce the capacity of a FLARE LUN or a MetaLUN. This capability exists only for hosts executing Microsoft Windows Server 2008 and later. In addition, it is a FLARE revision dependent feature. The LUN Shrink feature is available on FLARE revision 29.0 and later.

MetaLUN shrink allows for reducing the user capacity of a MetaLUN. It can reduce either striped or concatenated MetaLUNs by removing components or shrinking component LUNs within a MetaLUN.

When a MetaLUN is shrunk, components outside the shrunk capacity are removed from the MetaLUN. These component LUNs are unbound. In a striped component, each component LUN in the MetaLUN component is shrunk by the same amount. A component shrink does not re-locate data within the component. For a striped MetaLUN component, existing striping is maintained. Be aware that if a MetaLUN contains two or more MetaLUN components (which must be concatenated together) and the shrink total is more than the capacity of the last component, any other

components covered by the shrink amount are removed, and the component LUNs of the last remaining component are uniformly reduced.

Shrinking a MetaLUN may reduce the number of component LUNs in a striped component, which will have a subsequent effect on MetaLUN performance. After a LUN shrink is performed, a RAID group defragmentation may be performed on effected RAID groups to reclaim the freed capacity.

Review the online help in Navisphere for specific details on MetaLUN shrink.

Conclusion

MetaLUNs are the CLARiiON feature for scaling FLARE LUNs upward in capacity and performance.

MetaLUNs are a basic feature of CLARiiON. They leverage the CLARiiON's FLARE operating system, to provide on-the-fly LUN-expansion capabilities within the storage system. There is no additional hardware or appliance needed to support MetaLUNs. All of the CLARiiON layered applications for local and remote replication work with MetaLUNs.

A MetaLUN is the sum of its component LUNs in capacity, performance, and availability. Likewise, its component LUN's underlying RAID groups determine their capacity, performance, and availability. To get optimized capacity utilization, performance, and availability from a MetaLUN, you need to consider the MetaLUN's organization all the way to the drives at the RAID group level.

A MetaLUN is also a user-designed CLARiiON logical storage object. It can be designed for data storage capacity that exceeds available FLARE LUNs. It can be designed for performance that exceeds available FLARE LUNs. It can be designed for high capacity and performance, along with high availability. However, planning is involved to reach the best solution for these objectives. In addition, the future scaling of the MetaLUN needs to be carefully considered ahead of time.

MetaLUNs are not the only solution for a high-capacity or high-performance host LUN. Advances in storage technology have increased the performance and capacity of the CLARiiON's physical storage objects. FLARE LUNs have larger capacity and more IOPS than ever before. Large-capacity SATA drives and semi-conductor-based EFDs can be used in FLARE LUNs in lieu of creating a MetaLUN. In addition, features added to the CLARiiON further increase its performance, capacity utilization, and ease of use. Storage virtualization is one possibility. CLARiiON thin provisioning and the host-based LVM can also be used as substitutes for a MetaLUN. Thin provisioning is very easy to implement, and has a very low maintenance requirement. LVMs allow for scaling LUNs beyond the capability of a single storage processor maintained storage object.

MetaLUNs are managed through the Navisphere user interface. The Expand option is part of the LUN object menu. The expand option brings the user to the Expand Wizard, which guides users through the expansion process. Navisphere Analyzer is used to

get an in-depth understanding of how the MetaLUN and its entire set of component FLARE LUNs are performing. A set of Java-based secure CLI commands, which use the Navisphere 6.x security model, is available as an alternate means of managing MetaLUNs.

References

- *EMC CLARiiON Best Practices for Performance and Availability*
- *EMC CLARiiON Storage System Fundamentals*
- *Navisphere Command Line Interface (CLI) Version 6.X Reference for Basic, Access Logix, and MetaLUN Commands*
- *EMC Navisphere Manager Version 6.XX*