EMC VNX2 Deduplication and Compression
VNX5200, VNX5400, VNX5600, VNX5800, VNX7600, & VNX8000
Maximizing effective capacity utilization

Abstract
This white paper discusses the capacity optimization technologies delivered in the EMC® VNX™2 series of storage platforms. Deduplication and compression capabilities for file and block storage are delivered standard with the VNX Operating Environment.

March 2016
Copyright © 2016 EMC Corporation. All Rights Reserved.

EMC believes the information in this publication is accurate of its publication date. The information is subject to change without notice.

The information in this publication is provided “as is.” EMC Corporation makes no representations or warranties of any kind with respect to the information in this publication, and specifically disclaims implied warranties of merchantability or fitness for a particular purpose.

Use, copying, and distribution of any EMC software described in this publication requires an applicable software license.

For the most up to date listing of EMC product names, see EMC Corporation Trademarks on EMC.com.

VMware, VMware vCenter, and VMware View are registered trademarks or trademarks of VMware, Inc. in the United States and/or other jurisdictions. All other trademarks used herein are the property of their respective owners.

Part Number H12209.10
Table of Contents

Executive Summary .................................................................................................................. 5
Audience ................................................................................................................................. 5

Introduction to VNX data deduplication and compression ..................................................... 5

VNX Block Deduplication ....................................................................................................... 6
VNX Block Deduplication Overview ......................................................................................... 6
The VNX Block Deduplication Process ..................................................................................... 8
Management ........................................................................................................................... 12
  VNX Block Deduplication NaviSecCLI Commands ............................................................... 19
Block Deduplication with Multicore FAST Cache, FAST VP, and Multicore Cache ................. 19
Interoperability ......................................................................................................................... 21
Block Deduplication use cases ............................................................................................... 22
  Virtual Desktops and Virtual Servers .................................................................................... 22
  Common Data Environment ................................................................................................. 22
Performance and Capacity Savings ........................................................................................... 22

VNX Block Deduplication for VNX File .................................................................................... 23
Requirements and Supported Configurations ......................................................................... 23
Management ........................................................................................................................... 25
Considerations ......................................................................................................................... 32

VNX Block Compression .......................................................................................................... 33
VNX Block Compression Overview ......................................................................................... 33
The VNX Block Compression Process ..................................................................................... 33
  Compression Rate .................................................................................................................. 34
  Decompression operations ................................................................................................. 35
Management ........................................................................................................................... 35
  VNX Block Compression commands .................................................................................... 40
Available Limits ...................................................................................................................... 40
Interoperability ......................................................................................................................... 41
  Block Compression with VNX File ......................................................................................... 41

VNX File Deduplication and Compression ............................................................................... 41
VNX File Deduplication Overview .......................................................................................... 41
VNX File Compression Overview ............................................................................................ 42
The VNX File Deduplication and Compression Process ............................................................ 42
  Duplicate Detection Method .............................................................................................. 44
  Compression Method .......................................................................................................... 45
  CIFS Compression Enabled ................................................................................................. 46
Management ........................................................................................................................... 46
Executive Summary
Capacity-optimization technologies play a critical role in today's environment where companies need to do more with less. The EMC® VNX®2 series of storage systems are well-equipped to meet users' needs in this regard. By using deduplication and compression, users can significantly increase storage utilization for file and block data. The intelligent and automated deduplication and compression features on both Block and File are provided in the VNX Operating Environment (VNX OE) at no additional cost.

Management is simple and convenient. Once the capacity-optimization technologies are turned on, the system intelligently manages capacity-optimization processes as new data is written. With Unisphere®, users can manage block and file data from a single, user friendly interface.

This white paper discusses the capacity-optimization capabilities of VNX2 series systems, how they are best deployed, and how they fit in with other capacity-optimization technologies in the storage environment.

Audience
This white paper is intended for anyone interested in understanding the deduplication and compression functionality included with the VNX2 series storage systems.

Introduction to VNX data deduplication and compression
VNX systems are designed and built to handle the I/O demands of a large number of Flash drives. Performance-optimization features such as Multicore FAST Cache and Fully Automated Storage Tiering for Virtual Pools (FAST VP) move the busiest data onto the highest-performing drives, which helps to increase the system's IOPS-per-dollar.

The use of capacity efficiency features such as deduplication and compression play a large role in helping to lower the total cost of ownership. First, deduplication and compression both work to reduce the space requirements needed for datasets. Compounded efficiencies are attained when deduplication and compression are used with FAST VP and Multicore FAST Cache.

VNX File Deduplication and Compression achieve savings at the file level, while VNX Block Deduplication and VNX Block Compression achieve savings at the LUN level. All VNX deduplication and compression features discussed in this paper are available on the VNX2 Series (VNX5200, VNX5400, VNX5600, VNX5800, VNX7600, and VNX8000).

As part of EMC's effort to improve its products, EMC periodically releases revisions of its software. EMC recommends running the latest release available where possible. The product release notes provide the most up-to-date information on product features and enhancements.
VNX Block Deduplication

VNX Block Deduplication is a software feature included with the VNX2 series. In general, deduplication is the process of identifying duplicate data contained within a set of block storage objects and consolidating it such that only one actual copy of the data is used by many sources. This feature can result in significant space savings depending on the nature of the data. In the VNX2 series, VNX Block Deduplication utilizes the fixed block deduplication method with a set size of 8 KB to remove redundant data from a dataset. Block Deduplication is run post-process on the selected dataset.

VNX Block Deduplication Overview

The following diagrams illustrate the process VNX Block Deduplication uses to save space within a dataset. Figure 1 shows multiple Block storage based LUNs that have deduplication enabled on them. The deduplication process has not been run at this point in time.

Figure 1. VNX Block Deduplication example

Figure 2 shows 8 KB chunks of data that have been identified by the deduplication process as being exact copies of each other. This data will be deduplicated.

Figure 2
Figure 2. VNX Block Deduplication example

Figure 3 shows the process of deduplicating data on LUNs 1, 2, and 3. The first instance of each 8 KB chunk of data is kept in the pool, while the duplicates are deleted. Each LUN containing the chunks of data that were deduplicated will point to the copies that remain in the pool.

Figure 3. VNX Block Deduplication example

When a host writes to an address range that has previously been deduplicated to a shared block of storage, the system will send the write to a new location and that portion of the data will no longer be shared. The system will examine that data for possible deduplication on the next pass. Figure 4 shows this process.

Figure 4. Example of a write to deduplicated space
In the above example, the deduplication process removes the duplicates of the original of 8 KB chunks. Consider a storage array which stores multiple copies of the same data. In this scenario, the array needlessly copies much of the data repeatedly. Deduplication consolidates most of the 8KB duplicate parts and releases additional space for other purposes.

**The VNX Block Deduplication Process**

VNX Block Deduplication is enabled on a per pool LUN basis. For LUNs that will use Block Deduplication, EMC recommended that deduplication be enabled at the time of LUN creation. Creating a deduplication-enabled LUN will create the LUN directly within the deduplication container. When enabling deduplication on an existing LUN, the LUN will be migrated into the deduplication container using a background process. The data must reside in the container before the deduplication process can run on the dataset. During this migration, the deduplication state for the LUN will show as enabling and the migration progress can be viewed on the Deduplication tab within the LUN Properties window. Once the migration is complete, the LUN will be Thin and deduplication is enabled. To enable deduplication on a Classic LUN, first create a deduplication-enabled LUN within a pool and use LUN Migration to migrate the data to it.

For applications requiring consistent and predictable performance, EMC recommends using Thick LUNs. If Thin LUN performance is not acceptable, then do not use Block Deduplication.

During a migration of a pool LUN either into the deduplication container or out of the deduplication container when deduplication is disabled on a LUN, the pool requires free space greater than 110% of the consumed capacity of the original LUN. Creating a LUN with deduplication enabled is suggested, as this will eliminate the migration and required overhead that comes with it.

The deduplication container is private space within a pool in which all data for deduplication-enabled LUNs resides. This container is created when the first deduplication-enabled LUN is created, or deduplication is enabled on a LUN within a pool. The deduplication container is destroyed when deduplication is disabled on the last deduplicated LUN or when the LUN is deleted. There is only one deduplication container per storage pool. The total space used for the container depends directly on the amount of space used by the deduplicated LUNs. This space is not reserved when a pool is created, but rather is used when deduplicated LUNs exist. Non-deduplicated and deduplicated LUNs can coexist within the same pool.

When a deduplication container is created, the SP owning the container needs to be determined. The container owner is matched to the Allocation Owner of the first deduplicated LUN within the pool. For example, if the Allocation Owner of the LUN is SPA, then the deduplication container will be created on SPA. Enabling Block Deduplication on subsequent LUNs will cause the Allocation Owner of the LUNs to match the owner of the deduplication container. When enabling deduplication on existing LUNs, the Default and Current Owners will be retained after deduplication is
enabled, but the Allocation Owner will change to match the owner of the deduplication container.

A LUN’s Allocation Owner can be viewed on the General tab within the LUN Properties window in Unisphere. This is also where you determine which SP owns the pool’s deduplication container by viewing the Allocation Owner of a deduplication-enabled LUN on the pool.

EMC recommends that all LUNs with Block Deduplication enabled within the same pool should be owned by the same SP. After enabling deduplication on an existing LUN, verify that the Default Owner and Current Owner match the Allocation Owner of the LUN. If the Default Owner does not match the Allocation Owner, the Default Owner should be changed by using the Properties page of the LUN in Unisphere. If the LUN’s Current Owner does not match the Allocation Owner, the LUN should be trespassed by right-clicking the LUN in Unisphere and clicking Trespass. In VNX OE for Block version 05.33.006.5.096 and later, a warning is displayed when enabling deduplication on a LUN which resides on the peer SP of the deduplication container. This warning states that the ownership of the LUN will not be optimal, and recommends the user change the Default Owner to avoid performance impact.

When creating new LUNs with deduplication enabled, first determine the SP owner of the deduplication container by reviewing the Allocation Owner of a LUN with deduplication enabled within the pool. During the creation of the new LUNs with deduplication enabled in this pool, ensure that you select the Default Owner to match the SP owning the deduplication container for the pool in the Advanced tab within the Create LUN dialog window.

When creating deduplication-enabled LUNs in VNX OE for Block version 05.33.000.5.072 and later, the Default Owner is automatically matched to the owner of the deduplication container within the pool. This helps create deduplicated LUNs with an optimal ownership model. In VNX OE for Block version 05.33.006.5.096 and later, the optimal owner of deduplicated LUNs within the pool can be viewed in the Deduplication tab of the Pool Properties window in Unisphere, shown as the Optimal Deduplicated LUN SP Owner. The Optimal Deduplicated LUN SP Owner is also the SP which owns the deduplication container in the pool.

EMC also recommends balancing the deduplication containers between the SPs. Figure 5 shows an example of deduplication containers balanced across the Storage Processors of the system, with non-deduplication enabled LUNs on the peer SP of the deduplication containers. In this example, the deduplication container within Storage Pool 1 has been created on SPA. If enabling Block Deduplication on a second Storage Pool on the system, the deduplication container should be created on SPB, which balances the deduplication processes across the storage processors. To create the deduplication container on SPB, create a new deduplication enabled LUN with a Default Owner of SPB, or enable Block Deduplication on a LUN with an Allocation Owner of SPB within the second Storage Pool. The owner of a pool’s deduplication
container cannot be changed without disabling deduplication on all LUNs within a pool.

![Figure 5. Deduplication Container Balance Example](image)

VNX Block Deduplication runs in the background on the contents of a deduplication container post process, 12 hours after the previous deduplication run completed on the pool. Each pool in the system has an independent deduplication container and process unique to the pool. The contents in a deduplication container are compared against each other only.

When the 12-hour timer expires and the deduplication process is set to run, the number of running deduplication processes on the current SP is checked. A maximum of 3 deduplication processes per SP may run at a time, and if an open slot is not found, the deduplication process waits until one becomes available. The SP that the process runs on is based on the owner of the deduplication container.

Once a deduplication process is allowed to run, the data for the deduplicated LUNs is checked for a total of 64 GBs of new and updated data. If the pool does not contain 64 GBs of new or updated data, only the digest portion of the deduplication process is run and the twelve hour timer is reset. If 64 GBs of new data exists, the full deduplication process will run. The digest portion of the deduplication process is an algorithm that is run on each new 8 KB block of data on the deduplication container to create a unique digest (hash) which represents each piece of data. These digests are stored and later sorted on private space on the pool. The size of the private space consumed by deduplication depends directly on the number of digests created to represent the data inside the deduplication container. The deduplication process will run to completion, or until deduplication is paused either by the user or by the system.

During a deduplication process, all digests for the deduplicated LUNs in the pool are then compared to identify duplicates. Once duplicate digests are found, the blocks of data that the digests represent are bit-by-bit compared to verify the data is exactly the same. If the blocks of data are exactly the same, the index for each of the LUNs which contain the duplicate data will be updated to point to the oldest instance of the data. Once the indexes are updated, unused duplicate 8 KB blocks are removed. A 256MB
slice may be freed back to the pool for use when the slice no longer contains any data.

If a deduplication process has run on a pool for 4 hours straight, the run is paused and the system checks for deduplication processes waiting to run on that SP. If a deduplication process is waiting to run, it is allowed to run and the process that was paused will wait to continue. Queued processes are run in a round robin manner. Once a free slot opens, the paused process will continue where it left off. If there are no queued processes, the process continues to run.

In VNX OE for Block version 05.33.006.5.096 and later, the user may start a deduplication process on a pool at any time using the Force Deduplication option. This option bypasses the 64 GB of changed data requirement, and immediately starts the deduplication process if an available slot on the SP exists. Force Deduplication is best used when a large amount of data has changed on the pool and the user wishes to run deduplication, such as after a data migration. The amount of savings attained during a deduplication pass depends on the data that was changed since the last run. It is not recommended to run force deduplication often because the savings may be minimal. Force Deduplication is not a replacement for the normal deduplication process and should be used sparingly.

At any point in time, you may pause block deduplication at either the system level or the pool level. Pausing deduplication at either level prevents any deduplication process from running. Pausing deduplication will also pause any migrations on the system caused by enabling or disabling deduplication on a LUN. Once deduplication is resumed, either at the system level or pool level, the deduplication processes will check to see if a deduplication run is needed. Any in progress deduplication migrations will continue. If a deduplication process needs to run, the process on those pools will start shortly after the user resumes deduplication.

VNX OE for Block version 05.33.009.5.155 contains a number of enhancements to Block Deduplication. These improvements pertain to the core functionality of Block Deduplication, with specific updates to the data layout within the deduplication container and the deduplication process. In this release and later, the manner in which data is placed on the back-end storage has changed. For deduplication enabled LUNs, a stream detection process has been implemented to determine if incoming write data is sequential or random. This process will allocate storage and place data accordingly within the deduplication container.

Additionally this release makes a change to the deduplication process. To help improve performance and preserve the new data layout, block deduplication will make deduplication decisions based on the impact deduplication could potentially cause to the layout of the data. Deduplication will attempt to deduplicate longer contiguous regions of data in order to create more contiguous space for future writes. Also, host written zeros are accounted for in this deduplication method, but are ultimately unmapped rather than deduplicated.
Another improvement in the VNX OE for Block version 05.33.009.5.155 release has been made to the deduplication process and highly referenced blocks. When the deduplication process runs, there is a potential for a single 8 KB block to be referenced hundreds, or even thousands of times. This improvement has been made to not cause contention on a single block, by limiting the number of times a block can be referenced. In this instance, referencing another 8 KB copy of the data helps to reduce contention and improves performance of the system. Improvements to locking mechanisms when metadata updates are needed has also been made in an effort to improve performance.

In VNX OE for Block version 05.33.009.5.155 release and later, an enhancement has been made to the process of returning slices to the Storage Pool used by Block Deduplication. The number of slices locked during this process have been reduced, and the speed at which the evacuation process runs has been increased. These improvements help to shorten the amount of time an evacuation process takes, while also reducing the contention on slices with other features such as FAST VP.

After upgrading to VNX OE for Block version 05.33.009.5.155 release or later, existing implementations of Block Deduplication will receive the enhancements within this release. The new deduplication container layout however will only take effect on new writes to deduplication enabled LUNs on the system. In order to take full advantage of the new layout and the performance enhancements the new layout provides, existing LUNs must be migrated into a new deduplication container.

Management
You can easily manage VNX Block Deduplication by using either the Unisphere software or NaviSecCLI at the System, Pool, and LUN levels. Once enabled, the system automatically manages the processing of eliminating duplicate data based on the LUNs in the deduplication container.

Figure 6 shows the Create LUN dialog box. When creating a Pool LUN, the user is given the option of enabling VNX Block Deduplication at the time of creation. Notice that the Thin checkbox is also enabled, since Deduplicated LUNs are Thin LUNs by definition.
Figure 7 shows the Advanced tab of the Create LUN dialog box. When creating a Deduplicated LUN in VNX OE for Block version 05.33.000.5.072 and later, the Default Owner is automatically selected to match the owner of the deduplication container within the Pool if one exists. This is done to ensure proper LUN ownership within the deduplication container. The Auto choice is not selectable when a deduplication container exists in the Pool. The Default Owner will be Auto if no Deduplicated LUNs exist in the Pool, or the array is running an earlier version of VNX OE for Block. If running on a code previous to 05.33.000.5.072, you must manually set the Default Owner of the LUN to the SP that owns the deduplication container within the pool.
If the user changes the Default Owner to the peer SP in VNX OE for Block version 05.33.000.5.072 and later, the message shown in Figure 8 is displayed. By changing the Default Owner, the user is choosing to avoid the optimal ownership model for the LUN. As the message indicates, changing the owner may greatly impact performance.

![Figure 8: Deduplication ownership warning](image)

For Pool LUNs that are already created, controls for VNX Block Deduplication are available in the properties dialog box of the LUN. Figure 9 shows the Deduplication tab within the LUN properties window. The user is able to check the Turn on Deduplication check box to enable VNX Block Deduplication on the LUN.

![Figure 9: LUN properties Deduplication tab](image)

The user may also select one or more LUNs within the LUN list window in Unisphere. Right-click the LUN or LUNs, and select Turn On Deduplication to enable block deduplication. This is shown in Figure 10.
In VNX OE for Block version 05.33.008.5.119, the Deduplication tab within the LUN Properties dialog box has changed with the addition of Block Deduplication for VNX File. In this release and later, the user will no longer be able to enable or disable Block Deduplication on a LUN provided to VNX File. All Block Deduplication controls for VNX File LUNs is handled by VNX File. As shown in Figure 11, the Turn on Deduplication checkbox is not selectable, and a new Note is present.

When deduplication is enabled on any LUN, the system will automatically store the original LUN type, allocation SP, default SP, and FAST VP Policy. If deduplication is disabled on a LUN, that LUN is automatically migrated out of the deduplication container and the original properties are restored. If deduplication is disabled on a LUN which was created with deduplication enabled, the LUN will be Thin.

In VNX OE for Block version 05.33.006.5.096 and later, a warning is displayed when enabling deduplication on a LUN which resides on the peer SP of the deduplication container. This warning is shown in Figure 12. In this example, the deduplication container is owned by SPA, but the LUN has a Default, Current, and Allocation Owner
of SPB. After enabling deduplication on an existing LUN, the user should verify that the Default Owner and Current Owner match the Allocation Owner of the LUN. If the Default Owner does not match the Allocation Owner, the Default Owner should be changed by using the Properties page of the LUN in Unisphere. If the LUN’s Current Owner does not match the Allocation Owner, the LUN should be trespass by right-clicking the LUN in Unisphere and clicking Trespass.

![Figure 12. Enabling deduplication ownership warning](image1)

**Figure 12. Enabling deduplication ownership warning**

**Figure 13** shows the LUNs list found under the Storage tab in Unisphere. The window has been customized to show whether deduplication is enabled on a LUN. This is done by clicking the wrench icon in the top right of the LUNs window, selecting *Choose Columns*, and checking the *Deduplication* checkbox.

![Figure 13. LUNs window showing Deduplication column](image2)

**Figure 13. LUNs window showing Deduplication column**
Shortly after deduplication is enabled on a LUN, the LUN state will show as enabling in the Deduplication tab in the LUN properties window. While the LUN State shows as enabling, the LUN is migrating into the deduplication container within the pool.

At any time, the user can view deduplication information for a pool by navigating to the Deduplication tab within the Storage Pool Properties window (Figure 14).

![Figure 14. Deduplication tab within the Pool Properties window](image)

The system's Block Deduplication state can also be viewed at the top of the Deduplication tab within the Pool Properties window. The state can either be **On**, **Off**, or **Paused**. In the Properties box on the Deduplication tab, you can also view the current State of deduplication on this pool. The State will be **Idle**, **Pending**, **Running**, **Paused**, or **Faulted**. In the example above, the State is **Running** and the completion percentage and GB remaining values are displayed. The amount of data remaining is the amount of data needing to be digested. The process will show as running and 0 GBs remaining while digests are sorted, LUN indexes are updated, and duplicates are removed.

The **Pause** and **Resume** buttons for the Pool also reside within the Deduplication tab (Figure 14). Clicking the **Pause** button will prevent a deduplication pass from running or pause a running deduplication process on the Pool. Upon clicking **Resume**, a paused deduplication process will continue to run if the maximum processes for the SP are not exceeded or a deduplication process will be queued to run. The **Pause** and **Resume** buttons also control migrations due to enabling or disabling deduplication on a LUN. Clicking **Pause** or **Resume** will subsequently pause or resume any deduplication migrations on the Pool.
The Deduplication tab (Figure 14) is where you set the FAST VP Tiering Policy for all of the deduplicated LUNs in this pool. The data in the deduplication container is treated as a whole; thus the Tiering Policy applies to all deduplicated data on the pool. The same options for the Tiering Policy on non-deduplicated LUNs are available to the LUNs in the deduplication container. As with other pool-based LUNs, the default setting is Start High then Auto-Tier. If the FAST VP enabler is not installed on the system, the tiering policy will be a fixed value, and will reflect the Initial Tier data placement policy of the first deduplicated LUN on the pool.

Next on the Deduplication tab is the Deduplication rate. The Deduplication rate is the rate at which the deduplication process runs, not the level of deduplication performed. The default for this setting is Medium. The Deduplication rate also controls the rate at which deduplication migrations run. At a rate of medium, deduplication migrations caused by enabling or disabling deduplication on a LUN within the Pool will run at medium.

In VNX OE for Block version 05.33.006.5.096 and later, the Optimal Deduplicated LUN SP Owner is also displayed. This informs the user which SP the deduplicated LUNs within the pool should be owned on. This is also the owner of the deduplication container.

The Deduplicated LUNs Shared Capacity is displayed next on the Deduplication tab. This value, in GBs, is the total amount of space currently used by deduplication on the pool. This value includes space used by LUNs and Snapshots.

Deduplication and Snapshot Savings is also provided. This is the total amount of space, in GBs, the system is saving on this Pool by using deduplication. Assume that the same 10 GBs of application data is copied to four LUNs on the same pool. When these LUNs are deduplicated, only 1 copy of this data is needed, and the duplicates will be deleted. The Deduplication and Snapshot Savings in this example will be 30 GBs, as 10 GBs is kept for referencing and the other 3 instances of the 10 GBs of data is deleted. The savings may increase further if other duplicate data within the deduplication container can be identified.

The data for VNX Snapshots taken on deduplicated LUNs is also stored within the Deduplication Container. This allows the data to be deduplicated, and the savings attained are counted towards the pool savings.

Next on the Deduplication tab in Figure 14 is the Refresh button. The Refresh button at the bottom of the screen updates the information on the Deduplication tab when clicked.

In VNX OE for Block version 05.33.006.5.096 and later, the Force Deduplication button is also displayed. In Figure 14 the Force Deduplication button is shaded and not selectable because deduplication is already running on the pool. When selected, the 64 GB of changed data requirement is bypassed, and the deduplication process starts on the pool if an available slot on the SP exists. It is not recommended to force deduplication to run often, and it is not a replacement for the normal deduplication process.
Figure 15 is the Deduplication Summary window, which displays deduplication information for the array. In this window, the user is able to view the current System Deduplication State, which can either be shown as On, Off, or Paused. Next to the System Deduplication State are the Pause and Resume buttons. The user may Pause or Resume deduplication at the system level here.

In the Pools portion of the window, each pool that has deduplication-enabled LUNs is displayed. Information for each pool is displayed, including the Pool Name, the Deduplication Status (Pending, Running, or Paused), the selected Rate (High, Medium, or Low), the Total Capacity and Free Capacity for the pool, the Shared Capacity for the deduplicated LUNs, and Savings information for deduplication.

![Deduplication Summary Window](image)

**Figure 15. System level Deduplication Summary**

**VNX Block Deduplication NaviSecCLI Commands**

As with other Unisphere functions, the settings for VNX Block Deduplication can also be viewed or changed by using NaviSecCLI. To view information about deduplication or pause or resume deduplication at the system level, use the deduplication command. To enable or disable deduplication on a LUN use the lun -modify command. For pool level related statistics, use the storagepool command. For detailed information on each of these commands and others, please consult the *VNX Command Line Interface Reference for Block* documentation, located on EMC Online Support.

**Block Deduplication with Multicore FAST Cache, FAST VP, and Multicore Cache**

The basic purpose of VNX Block Deduplication is to eliminate data that is common between deduplication enabled LUNs within a pool. While saving space in itself is a good reason to enable deduplication, EMC suggests using deduplication with Multicore FAST Cache and FAST VP to achieve better efficiency of these features.
After duplicate data has been removed within a deduplication container, FAST VP operations become more efficient as there is less data to track and promote. Also, with a consolidated storage footprint, FAST VP can better use space on the pool. Consider multiple LUNs on the same pool that have been deduplicated. Before deduplication, due to application activity on these LUNs, the slices for each of these LUNs would be contending with each other for promotion. After deduplication, exact copies of data are removed and contention between common data is eliminated. Now, only unique data within the deduplication container is contending with the data in the pool for promotion.

Multicore FAST Cache also is more efficient when used with deduplication. With the removal of exact copies of active data on the underlying pool, Multicore FAST Cache can also free this space for other promotions. Data not previously considered for promotion may also benefit due to the consolidation of I/O the deduplication causes. Consider multiple copies of an 8 KB block, each with minimal activity. After deduplication runs and removes all but one 8 KB instance of this data, all I/O to each of the previous blocks is directed to the single 8 KB block. This block becomes more utilized which may cause a promotion and allow the data to benefit from residing on Flash based storage.

Multicore Cache also benefits when hot 8 KB blocks are deduplicated. Deduplication of these hot blocks frees memory in Multicore Cache for other hot blocks to use. If deduplication causes block to be highly accessed, the blocks may stay within Multicore Cache.
### Interoperability

Table 1 VNX Block Deduplication Interoperability

<table>
<thead>
<tr>
<th>Optional Feature</th>
<th>Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>VNX File</td>
<td>Fully supported</td>
</tr>
<tr>
<td>LUN Migration</td>
<td>Fully supported in or out of the deduplicated Pool, but not while a LUN is in “enabling” or “disabling” deduplication states (because that uses a LUN migration internally, and you cannot initiate on a LUN that is already in the midst of migrating). Similarly, a LUN that is currently migrating cannot have its deduplication-enabled state changed.</td>
</tr>
<tr>
<td>VNX Snapshots</td>
<td>Fully supported. Cannot take a snapshot while the LUN is being migrated in or out of the pool (when the LUN is in “enabling” or “disabling” states). Snapshots that exist prior to enabling or disabling deduplication will be destroyed when the operation completes (with a user warning before proceeding).</td>
</tr>
<tr>
<td>Compression</td>
<td>Not supported because the current implementation of compression is not compatible with the Deduplication container; thus each efficiency feature would counter-act each other.</td>
</tr>
<tr>
<td>Controller Based Encryption</td>
<td>Fully supported</td>
</tr>
<tr>
<td>FAST VP</td>
<td>Fully supported, but with a single set of attributes on the pool applying to all deduplication-enabled LUNs in that pool. (Non deduplicated LUNs still maintain per-LUN FAST VP policy).</td>
</tr>
<tr>
<td>SnapView Snapshots</td>
<td>Fully supported</td>
</tr>
<tr>
<td>SnapView Clones</td>
<td>Fully supported; however best practices dictate that the clone should not reside in the same pool as the source LUN. In the case of deduplication, if the clone is in the same deduplication pool, it will share duplicate blocks thus defeating the purpose of creating a clone of the source on separate spindles. Unisphere will issue a warning if the user creates a clone target in the same pool as the source LUN.</td>
</tr>
<tr>
<td>MirrorView (Sync &amp; Async)</td>
<td>Fully supported</td>
</tr>
<tr>
<td>SAN Copy</td>
<td>Fully supported</td>
</tr>
<tr>
<td>Reserved LUN Pool</td>
<td>Deduplicated LUN cannot be used in the Reserved LUN Pool</td>
</tr>
<tr>
<td>Clone Private LUN</td>
<td>Deduplicated LUN cannot be used in the Clone Private LUN</td>
</tr>
<tr>
<td>Write Intent Log</td>
<td>Deduplicated LUN cannot be used in the Write Intent Log</td>
</tr>
<tr>
<td>Private LUN</td>
<td>In general, Deduplicated LUNs cannot be private</td>
</tr>
</tbody>
</table>
**Block Deduplication use cases**
Clearly deduplication is most efficient when the storage environment includes multiple copies of the exact same data. This is especially true for data sets which contain a large volume of read-only, non-changing files. Below are the most prevalent use cases for deduplication.

**Virtual Desktops and Virtual Servers**
Virtual desktops and servers are a popular choice to reduce the physical hardware needs for an environment. Virtual environments typically contain common base images on which applications are loaded. Base images typically have files that may remain unchanged and will thus be common with other base images of that type. This environment of common data is ideal for Block Deduplication, as it will be able to free up the duplicate data. The savings ratio increases as more desktops with common data are deployed. The effectiveness of saving space by deduplication directly depends on the amount of duplicate data.

**Common Data Environment**
Deduplication is also useful in environments where multiple copies of the same LUNs exist. Whether these LUNs are production LUNs with common data or backups of a common source, large savings can be achieved when deduplication is used. Various customers have created test and development copies of multi-LUN data sets and wish to reduce their storage investment, by leveraging deduplication. Block Deduplication is also efficient when large amounts of common data exists within a LUN.

**Performance and Capacity Savings**
While Block Deduplication can help achieve large amounts of space savings, it is suggested to test Block Deduplication before enabling it in production. In the instance where Block Deduplication is enabled on a data set which does not contain a large amount of duplicate data, the user may choose to turn it off because removing the additional overhead may outweigh the space savings achieved. Leaving Block Deduplication disabled on response time sensitive applications may also be desirable.

The VNX2 product line is highly efficient at utilizing Flash drives to their full performance potential. As the number of Flash drives increase and larger portions of the workload is served out of Flash, the system utilizes more CPU resources to maximize that potential. Block Deduplication is a data service that requires additional overhead to the normal code path. When considering Block Deduplication, it is recommended to review the Utilization of the system and ensure CPU overhead is available. In instances where portions of data on a single LUN are a good fit for Block Deduplication while other portions are not, consider placing the data on different LUNs when possible.
VNX Block Deduplication for VNX File

In VNX OE for Block version 05.33.008.5.119 and VNX OE for File 8.1.8.119 VNX Block Deduplication can be used with VNX File. Enabling VNX Block Deduplication for a VNX File Mapped Storage Pool or Meta Volume based File System causes all underlying LUNs to become deduplication enabled thin LUNs. VNX Block Deduplication utilizes an 8 KB Fixed-Block Deduplication method of removing redundant data from all deduplication enabled LUNs within a Block Storage Pool. Deduplication savings with VNX Block Deduplication are returned to the deduplication enabled LUNs or the Block Storage Pool. For more information on VNX Block Deduplication, including an overview of the deduplication process, please review the VNX Block Deduplication section found within this paper.

When enabling VNX Block Deduplication for VNX File, all LUNs become deduplication enabled and migrate in the Storage Pool's deduplication container. If the underlying Block Storage Pool is shared between block-only LUNs and VNX File LUNs and a container exists, the VNX File LUNs will migrate into the existing container. If no deduplication container exists, one will be created and the migrations are started. As a Best Practice, a system's deduplication containers should be evenly spread across each SP. To control the owner of the deduplication container when using VNX Block Deduplication for VNX File, the user should create a small deduplication enabled LUN within the Storage Pool with a Default Owner matching the SP they want the deduplication container on. At this point, the user can enable VNX Block Deduplication for VNX File and delete the temporary LUN. For more information on the deduplication container and best practices, please consult the section named The VNX Block Deduplication Process and Figure 5.

While VNX Block Deduplication for VNX File and VNX File Deduplication and Compression can be enabled on the same resources, EMC strongly recommends only enabling one deduplication feature. Before deciding on a deduplication feature to use with VNX File, many considerations need to be investigated. Common considerations include the level of performance the VNX File resources require, should deduplication target active data, at what level should the deduplication savings be returned to, and what deduplication method would the current layout most benefit from. These considerations will help determine which deduplication method is best for the VNX File dataset. Appendix C: Comparing VNX Block Deduplication for VNX File Data and VNX File Deduplication and Compression outlines the major differences between the deduplication methods for VNX File Data.

Requirements and Supported Configurations
A number of requirements on VNX File need to be met before enabling VNX Block Deduplication on a VNX File resource. If these requirements are not met, a warning appears and deduplication is not enabled.

- Enabling VNX Block Deduplication is only supported on Mapped Storage Pools or Meta Volume based File Systems and Checkpoint Storage. The underlying block storage for these configurations must be created on Block Storage Pool
LUNs. VNX Block Deduplication can be enabled on thick or thin LUNs within a Storage Pool.

- Enabling VNX Block Deduplication on RAID Group based (Classic) LUNs is not supported.

- If a File System currently created on Classic LUNs is a good candidate for VNX Block Deduplication, it must migrated to a configuration based on Block Storage Pool LUNs before VNX Block Deduplication for VNX File can be enabled.

- VNX Block Deduplication for VNX File can only be enabled on configurations which do not include shared disks. A shared disk is a Block Storage Pool LUN which has been partitioned into multiple slices on VNX File and added to multiple resources. Figure 16 shows two examples of shared disks. On the left, LUN 2 has been shared between two File Pools. On the right, LUN 2 has been shared between Meta Volume 1 and Meta Volume 2. As VNX Block Deduplication is only enabled on whole LUNs, enabling deduplication on a shared disk is not supported.

![Figure 16. Shared Disk Example.](image)

- When choosing to extend a Meta Volume or File Pool in which deduplication is enabled, only expanding the resource with whole disk volumes is supported. When the extension is started, the new LUN will inherit the deduplication state of the Meta Volume or File Pool. In this instance, VNX Block Deduplication will migrate the LUN, which can be thick or thin, into the deduplication container automatically.

- VNX Block Deduplication for VNX File also requires that all File Systems utilize log type split. Split log technology allows a single File System and its log be unmounted from a Data Mover independently of other File Systems on the system if the need arises. Split log is the default log type in VNX OE for File version 8.1.6.96 and later, and is strongly recommended when utilizing thin LUNs as the underlying storage for VNX File resources. If a File System built on a Storage Pool utilizes a log type of common, VNX Block Deduplication cannot be enabled.
Management

VNX Block Deduplication is typically enabled on Block Storage Pool LUNs directly via Unisphere or NaviSecCLI. In VNX OE for Block version 05.33.008.5.119 and VNX OE for File 8.1.8.119, user controls for Block Deduplication have been disabled for LUNs provided to VNX File. Controlling of VNX Block Deduplication for VNX File LUNs is handled by VNX File. As shown in Figure 17, the Turn on Deduplication checkbox is not selectable within the Deduplication tab within the LUN Properties window, and a new Note is present. If VNX Block Deduplication enabled LUNs are provided to VNX File, the current LUN deduplication state will be ignored and will be updated if needed depending on the state on VNX File.

![Figure 17. LUN properties Deduplication tab for a VNX File LUN](image)

To enable or disable VNX Block Deduplication on a Storage Pool for File, view the Storage Pool Properties window. Within the Storage Pool Properties window is an option named Fixed-Block Deduplication Enabled, which can be seen in Figure 18. To enable or disable VNX Block Deduplication for VNX File on this Mapped Storage Pool, select or deselect the checkbox next to Fixed-Block Deduplication Enabled and click apply. When changing the state of this checkbox, VNX File determines which disk volumes are currently in use by File Systems created on the Mapped Storage Pool. These disk volumes are then sent commands to either enable or disable VNX Block Deduplication. The user may also choose to use Control Station CLI via the `nas_pool -modify -fixed_block_dedupe` command to enabled or disable VNX Block Deduplication for VNX File on a Mapped Storage Pool.
After enabling or disabling VNX Block Deduplication, a migration into or out of the deduplication container begins. To view which LUNs are migrating, view the LUNs within the Block Storage Pool, as shown in Figure 19. If the Deduplication column is not shown, click the wrench icon in the LUNs window and select Choose Columns.

Also shown in Figure 19 is a single LUN which is enabling Block Deduplication. In this example, VNX Block Deduplication should be enabling on all 10 LUNs within the Storage Pool based on their usage within the Mapped Storage Pool. However, due to the minimal Free Capacity within this Block Storage Pool, only 1 enabling process was allowed to start. When enabling VNX Block Deduplication each migration requires free space greater than 110% of the consumed capacity of the original LUN. Once the migration completes, the source LUN is deleted and space is freed back to the Storage Pool.

VNX Block Deduplication for VNX File contains an internal process which checks the Fixed-Block Deduplication states every 30 minutes. During this process, all back-end storage resources are also checked for consistency with the current state. If a back-end LUN is in the wrong deduplication state, the state will be corrected, which will require an enabling or disabling process to start.
Figure 19. VNX Block Deduplication for VNX File. Deduplication enabling.

Figure 20 below is displayed when deduplication enabled LUNs for VNX File are not currently on their optimal owners. This message is common when enabling deduplication on multiple LUNs for VNX File. For VNX Block Deduplication enabled LUNs, optimal ownership is achieved when the Default and Current Owners of all deduplication enabled LUNs within a pool match the deduplication container owner within the Block Storage Pool. The LUN numbers for the affected LUNs is mentioned in the error.

![Query storage pools. Some Fixed-Block Deduplicated enabled LUNs are not associated with their optimal owners. Non-optimal ownership of Fixed-Block Deduplication enabled LUNs may impact performance. IDs of affected backend luns: 5,7,3,11... Details...]

Figure 20. Deduplicated LUNs are not associated with their optimal owners error.

To correct the ownership of VNX File LUNs, you must first determine the Optimal Deduplicated LUN SP Owner for the underlying Block Storage Pool. Figure 14 shows the location of the Optimal Deduplicated LUN SP Owner within the Deduplication tab of the Storage Pool Properties window. Once the optimal owner is determined, the user must manually change the Default Owner, shown in Figure 12, on each of the affected LUNs and issue `nas_storage -check -all` and `nas_storage -failback` commands via Control Station CLI.

Figure 21 shows an example error when VNX Block Deduplication cannot be enabled on a LUN within a Storage Pool due to the Free Capacity of the Storage Pool. This message indicates to extend the pool to increase the Free Capacity. If migrations currently exist, allowing them to finish will free space within the Storage Pool and allow the enabling process to continue. This message will clear from the Mapped Pool Properties window once deduplication is enabled or disabled on all LUNs.
Figure 21. Could not enabled VNX Block Deduplication error.

Figure 22 below shows an error seen while deduplication is enabling or disabling on the underlying LUNs. The error informs the user that not all LUNs have a deduplication state which matches the setting on VNX File. Listed in this error are the back-end LUN numbers affected. This can be ignored if LUNs are still migrating due to enabling or disabling deduplication.

Figure 22. LUNs in incorrect deduplication state error.

Figure 23 shows the Properties window for a File System created on a Mapped Storage Pool. As VNX Block Deduplication is enabled at the Mapped Storage Pool level, **Fixed-Block Deduplication Enabled** reflects the state of the Mapped Storage Pool. In this instance, **Fixed-Block Deduplication Enabled** says **Yes** as VNX Block Deduplication is enabled for the Storage Pool.

![Figure 23. File System on Mapped Storage Pool. Properties Window.](image)

Figure 24 shows an example of creating a File System on a Meta Volume. During the creation process, the user can choose to enable VNX Deduplication for VNX File by checking the **Fixed-Block Deduplication Enabled** checkbox. The Meta Volume in this
example was previously creating using two disk volumes. Using the Control Station CLI command `nas_fs -create` with the `-fixed_block_dedup` option allows the user to specify if block deduplication is enabled or disabled when creating a File System on a Meta Volume.

![Figure 24. File System on Meta Volume Creation Window.](image)

**Figure 24.** File System on Meta Volume Creation Window.

**Figure 25** shows the properties window of a File System created on a Meta Volume. VNX Block Deduplication can only be enabled and disabled at the File System level when the File System is created on a Meta Volume. The user is also able to enable or disable VNX Block Deduplication on an existing File System built on a Meta Volume using the `-fixed_block_dedupe` option with the `nas_fs -modify` Control Station CLI command.
Figure 25. File System on Meta Volume Properties Window.

Figure 26 shows an example in which the Checkpoint Storage is being configured on a Meta Volume. At time of creation, the user is able to enable VNX Block Deduplication using the Fixed-Block Deduplication Enabled option. When VNX Block Deduplication is enabled on the Checkpoint Storage, Auto Extend is disabled. To extend the Checkpoint Storage the user must manually extend it using a Meta Volume utilizing whole disk volumes. If the Checkpoint Storage is created from a Storage Pool, Fixed-Block Deduplication Enabled will state either yes or no as the Storage Pool’s VNX Block Deduplication state is inherited by the Checkpoint Storage.

Figure 26. Configure Checkpoint Storage on Meta Volume Example.
Figure 27. shows the **Enter File System Info** step within the **New File System Wizard**. When using the wizard to create a File System on a Mapped Storage Pool, the user will be presented with the state of Fixed-Block Deduplication for the Mapped Storage Pool. As VNX Block Deduplication is enabled at the Mapped Storage Pool level, the state will be **Yes** for enabled, and **No** for disabled.

Figure 27. New File System Wizard. Create File System on Storage Pool.

Figure 28. shows the **Enter File System Info** step within the **New File System Wizard**. When using the wizard to create a File System on a Meta Volume, the user is able to select the checkbox next to **Fixed-Block Deduplication Enabled** to enable VNX Block Deduplication at time of creation.

Figure 28. New File System Wizard. Create File System on Meta Volume.
Figure 29 shows the Deduplication tab within the Block Storage Pool Properties window. When VNX Block Deduplication is enabled for VNX File, the Deduplication and Snapshot Savings reported for the Storage Pool is how much space has been saved using VNX Block Deduplication. The savings reported are for the deduplication container, which is common for block and file LUNs. When using VNX Block Deduplication, savings are returned to the deduplication container directly, or the Block Storage Pool if enough space is freed.

Considerations
For applications requiring consistent and predictable performance, EMC strongly recommends that you create File Systems using thick LUNs with Block Deduplication disabled. In VNX OE for Block Release 33, the default LUN type for a new LUN is thin; this can be accomplished by unchecking the ‘Thin’ checkbox when creating the LUN. Block Deduplication will be disabled by default. Deduplication and compression optimization can be achieved using the thin and File deduplication (which includes compression) attributes at the File System level. For more information on the major differences VNX Block Deduplication and VNX File Deduplication and Compression, please see Appendix C: Comparing VNX Block Deduplication for VNX File Data and VNX File Deduplication and Compression.
VNX Block Compression

VNX Block Compression Overview
In general, compression is a process that attempts to reduce the total space used by a dataset. VNX Block Compression works in 64 KB increments to reduce the storage footprint of a dataset and provide savings to the user. As accessing compressed data may cause a decompression operation before the I/O is completed, compression is not suggested to be used on active datasets.

The VNX Block Compression Process
VNX Block Compression can be enabled on any LUN type on the system. Once compression is enabled on a pool-based LUN, an initial compression process is run and the LUN becomes Thin. If Thin LUN performance is not acceptable, then Block Compression should not be used. If VNX Block Compression is enabled on a Classic LUN, this will cause a LUN migration to a pool-based LUN to occur. Thin LUNs are used because the Thin LUN technology allows the compression process to return freed slices to the pool. Capacity is allocated from a pool to LUNs in 256 MB slices. Over time, if enough 8 KB blocks are freed by compression, a slice can be returned to the pool for use by any LUN in the pool.

VNX Block Compression processes data in 64 KB increments. The speed at which the compression operation runs can be controlled by the Compression Rate setting (High, Medium, or Low). This value, explained later in this section, is set on a per-LUN basis and controls the rate at which data is processed by the compression operation. Compression can also be paused at the system and LUN level. When compression is paused, no new chunks of data will be processed to save space. Selecting resume will allow the compression process to continue.

VNX Block Compression will only compress data if at least 8 KB of the 64 KB can be saved. If the resulting savings from compression is less than 8 KB within the 64 KB chunk, the data is written to the LUN uncompressed. The VNX will not compress if there is not sufficient savings, which shows the efficiency of the product.

While the compression process is running, 64 KB chunks of data are processed to save space where possible. Once the chunk of data is compressed, I/Os to the compressed data will cause the data to decompress before the I/O can be completed. Block data compression is intended for relatively inactive data that requires the high availability of the VNX system as reading or updating compressed data will cause decompression of the data first. This decompression operation is additional overhead that compression causes. Consider static data repositories or copies of active data sets that users want to keep on highly available storage.

Disabling compression on a LUN will caused all compressed data for the LUN to decompress. This will increase the size of the LUN on disk, so space considerations should be taken into account. Decompression operations are further explained later in this section.
Compression Rate
Moving datasets from Classic LUNs or Thick LUNs to Thin LUNs provides the benefits of recapturing and repurposing unused capacity and provides a “pay as you go” capacity consumption model. Users make this tradeoff when they do not have stringent IOPs and response time requirements. Compression takes this a step further, providing even greater capacity savings with the tradeoff of additional system resources and LUN performance. The user may adjust the compression rate in the LUN Properties window at any time. The choices are Low, Medium, or High, with Medium being the default rate. Compression is best suited for data that is largely inactive but requires five 9s availability.

EMC recommends that you change the compression rate to Low at the system level when response-time critical applications are running on the storage system. You can also Pause compression at the system level during spikes in activity if you wish. The process of returning capacity to the pool that has been freed by compression can contribute to CPU and write cache usage as data is consolidated onto as few slices as possible. This process can also continue after the compression process completes. Pausing the compression feature will ensure any background compression or associated space reclamation operations do not impact server I/O.

The initial compression of Classic LUNs closely tracks the behavior of standard LUN Migration to a Thin LUN target. Compression is initially performed inline with the migration.

The difference in bandwidth with different compression rate settings for the new VNX series is shown in Figure 30. There is a larger difference between the low and medium rates than between medium and high. This behavior is consistent across all models.

Figure 30. Compression Rates
The compressibility of the data has little bearing on compression achieved during compression. When data is highly compressible (8:1 compression rate), the compression bandwidth may be 10 percent lower than is the case when compressing
moderately compressible data (1.5:1). Differences are only notable when approaching the max allowable compression operations with the rate set to High. The two cases are equal when the rate setting is Medium or Low.

Impact on server I/O can be moderate to high when compared to the performance of non-compressed Thin LUNs. The inline operations, inherent to reads and writes of compressed data, affect the performance of individual I/O threads; therefore, we do not recommend compression for LUNs with I/O-intensive or response-time-sensitive applications on them.

Compression's strength is improved capacity utilization. Therefore, compression is not recommended for active database or messaging systems, but it can successfully be applied to more static datasets like archives, clones of database, and messaging-system volumes. The best use cases are those where data needs to be stored most efficiently and with a high degree of availability.

**Decompression operations**

The decompression process restores compressed data to its original size. When compression is disabled on a compressed LUN, the entire LUN is processed in the background. When the decompression process completes, LUNs that were formerly Classic LUNs and Thick LUNs will be fully-allocated Thin LUNs within the pool with the same size as the original LUN. Thin LUNs are migrated to a new Thin LUN when compression is disabled.

If the pool becomes about 91 percent full during decompression, the compression state of the LUN will become **system paused**. Host I/O can continue to the LUN uninterrupted while compression is in this state. This behavior is implemented as a safeguard to ensure there is pool capacity available for new host I/O. If the user decides to continue decompression without adding capacity to the pool, the user can manually override this safeguard and resume decompression. However, the compression state will again become **system paused** if the pool reaches 98 percent full. This safeguard cannot be overridden.

LUN Migration offers an alternative to decompression. This option might be chosen when a change in LUN type may be warranted and/or different physical spindles may be used. Users can migrate a compressed LUN to a Classic LUN or a LUN in another pool, for example, as long as the target LUN’s user capacity is the same or larger than the compressed LUN’s user capacity.

**Management**

An online LUN migration of Classic LUNs to Thin LUNs is performed when compression is enabled on a Classic LUN. All aspects of the migration are handled for the user with no interruption to server I/O. Since Classic LUNs do not already reside in a pool, the user is prompted to select a destination pool for the compressed LUN. Figure 31 shows the **Turn On Compression** dialog box that allows the user to select an existing pool or launch the **Create Pool** dialog box to create a new destination pool. Users can also set the rate of the migration in this dialog box. Note that this is for migration only. It is not the in place conversion that takes place when LUNs are already in the pool.
Figure 31. Destination Pool Selection for Classic LUN Compression

All currently eligible pools are listed in the Pool drop-down menu. Pools are only shown if they have enough free capacity to accommodate the user capacity of the Classic LUN. Capacity equal to the user capacity of the Classic LUN is reserved in the pool to ensure the process can complete. After the migration is complete, the original Classic LUN is unbound and its capacity is available to create new LUNs in the RAID group.

Figure 32 below shows a Thick LUN before enabling compression. The user capacity of the LUN is 250 GBs, but 257 GBs are consumed in the pool since Thick LUNs consume capacity equal to the user capacity plus metadata. After compressing the LUN, users will see the consumed capacity of both the LUN and pool decrease. Consumed capacity in this case is reduced due to savings from the LUN being Thin and compressed.

Figure 32. Thick LUN properties before compression

VNX Block Compression is enabled by checking the Turn on Compression checkbox and clicking Apply, as shown in Figure 33. After this LUN is in a Compressed state (Figure 33), the LUN only consumes 63 GBs of space in the pool, for a savings of just over 193 GBs.
This savings represents the benefits of both moving the LUN from a Thick LUN to a Thin LUN and compressing the data. Space savings attained after compression is enabled on a Thick LUN in a pool entirely depends on the free space on the LUN and the compressibility of the data on the LUN.

The compression rate, shown above in Figure 33, determines how aggressively compression is performed (it is not the level of data reduction). The compression rate applies to initial compression, subsequent compression of new data, and decompression operations. The options are High, Medium, and Low, with the default rate of Medium; you can change the rate at any time. You may also Pause or Resume compression on the Compression tab of the LUN properties window.

Figure 34 shows the Storage Pool Properties window before the LUN was compressed. As you can see the pool is 9.53% full.
Figure 34. Pool Properties before compression

Figure 35 shows the Pool Properties after compression of the 250 GB LUN. The Percent Full dropped from 9.53% to 8.058% and the free space increased from 11575.564 GB to 11763.932 GB. The capacity saved from compressing the LUN is returned to the pool for use by any LUN in the pool. Server-visible metrics like User Capacity, Subscribed Capacity, and Percent Subscribed remain unchanged.
To view the LUN compression states, see Unisphere Help for a list and more information about each state. The table located in the Help shows the compression state, a description of that state, and what LUN types this state is valid for.

The **Compressed LUNs Summary** dialog box shown in Figure 36 provides a consolidated view of block compression activity for all LUNs. It also provides system-level compression control by using the **Pause Feature** button and **Resume Feature** button at the bottom of the dialog box.

![Figure 36. Compressed LUNs Summary Dialog Box](image)

When using the **Pause Feature** option, all LUN-level compression operations are paused. Compression operations occurring by using LUN Migration can be paused in the same way as other Compression operations. These operations can be cancelled in the **Compression** tab in the **LUN Properties** dialog box.

**Figure 37** shows the LUN table of a VNX system in the Unisphere software. Users can configure optional columns for the LUN attributes Thin and Compression.
VNX Block Compression commands
As with other Unisphere functions, the settings for VNX Block Compression can also be viewed or changed by using NaviSecCLI. The command used for compression is aptly named `compression`. For a detailed synopsis of this command and others, please consult the *VNX Command Line Interface Reference for Block* documentation, located on EMC Online Support.

Available Limits

**Table 2 Limits for the VNX Block Data Compression Feature**

<table>
<thead>
<tr>
<th>VNX Model</th>
<th>5200/5400/5600</th>
<th>5800</th>
<th>7600</th>
<th>8000</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Compressed LUNs</td>
<td>1000</td>
<td>2000</td>
<td>3000</td>
<td>4000</td>
</tr>
<tr>
<td>Concurrent Compressions per SP</td>
<td>20</td>
<td>20</td>
<td>32</td>
<td>40</td>
</tr>
<tr>
<td>Concurrent migrations per array</td>
<td>16</td>
<td>16</td>
<td>24</td>
<td>24</td>
</tr>
</tbody>
</table>

**Notes:**

- Concurrent compression operations include initial compression of data on Thin LUNs, compression of new data, and decompressions.
- Concurrent migrations include any initial compression of Classic LUNs where the LUN being compressed must be migrated to a Thin LUN. (This also includes decompressions.)
- Compression and migration limits are not dependent on each other.
Interoperability

Table 3 VNX Block Compression Interoperability

<table>
<thead>
<tr>
<th>Optional Feature</th>
<th>Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>LUN Migration</td>
<td>VNX Block Compression cannot be enabled on a LUN while the LUN is being migrated.</td>
</tr>
<tr>
<td>VNX Snapshots</td>
<td>Cannot be compressed with VNX Block Compression.</td>
</tr>
<tr>
<td>Deduplicated LUN</td>
<td>Not supported with VNX Block Compression.</td>
</tr>
<tr>
<td>Controller Based Encryption</td>
<td>Fully Supported</td>
</tr>
<tr>
<td>FAST VP</td>
<td>Fully Supported</td>
</tr>
<tr>
<td>SnapView Snapshots</td>
<td>Not supported with VNX Block Compression.</td>
</tr>
<tr>
<td>SnapView Clones</td>
<td>A VNX Block Compression enabled LUN can be a clone source.</td>
</tr>
<tr>
<td>MirrorView (Sync &amp; Async)</td>
<td>A VNX Block Compression enabled LUN cannot be replicated to a pre-29 FLARE code array.</td>
</tr>
<tr>
<td>SAN Copy</td>
<td>A VNX Block Compression enabled LUN can be the destination of a San Copy Session. All data is treated as new, thus a decompression and re-compression operation may occur for the updates.</td>
</tr>
<tr>
<td>Reserved LUN Pool</td>
<td>A VNX Block Compression enabled LUN cannot be used in the Reserved LUN Pool.</td>
</tr>
<tr>
<td>Clone Private LUN</td>
<td>A VNX Block Compression enabled LUN cannot be used as a Clone Private LUN.</td>
</tr>
<tr>
<td>Write Intent Log</td>
<td>A VNX Block Compression enabled LUN cannot be used as the Write Intent Log</td>
</tr>
<tr>
<td>Private LUN</td>
<td>Not supported with VNX Block Compression.</td>
</tr>
</tbody>
</table>

Block Compression with VNX File

EMC strongly recommends that you create File Systems using thick, non-compressed LUNs. In VNX OE for Block Release 33, the default LUN type for a new LUN is thin so this can be accomplished by unchecking the ‘Thin’ checkbox when creating the LUN. Block Compression will be disabled by default. Compression optimization can be achieved using the thin and File deduplication (which includes compression) attributes at the File System level.

VNX File Deduplication and Compression

VNX File Deduplication Overview

When enabled on a File System, VNX File Deduplication works to remove duplicate files found within a common file system. This deduplication technique is referred to as Single-Instance Storage (SIS). SIS is not an EMC specific deduplication technique, but rather is one of the many methods of data deduplication. By removing duplicate files, the amount of space needed for storage is reduced. The savings achieved while
using File Deduplication will vary as the savings entirely depend on the similarity of the files being stored on the file system.

**VNX File Compression Overview**

When enabled on a File System, VNX File Compression works to reduce the consumed size of files on disk. File size reduction is achieved by examining the contents of a file on a file system and eliminating the common areas. The amount of savings due to the compression process entirely depends on the contents of each file. Files with many repetitive strings achieve the highest level of savings. Unlike with deduplication, compression can attain space savings within unique and duplicate files.

**The VNX File Deduplication and Compression Process**

When enabled on a file system, VNX File Deduplication and Compression periodically scans the File System for files that match the policy criteria (explained below). Once files that match the policy criteria are identified, the files are deduplicated using sha1 or byte method if Deduplication is enabled. After deduplication has run, or skipped in the case that Deduplication is disabled, compression runs using the Fast or Deep method and attempts to reduce the overall space used by the files. The user can choose between multiple methods of deduplication and compression, both of which are explained later in this section.

*Table 4* below contains the VNX File Deduplication and Compression settings common to the File System and Data Mover. Each of the following settings must be satisfied for deduplication to run. These settings are found in the **File System Properties** window and the **Data Mover Deduplication** window.
### Table 4 VNX File Deduplication and Compression File System/Data Mover Settings

<table>
<thead>
<tr>
<th>Setting</th>
<th>Definition</th>
<th>Default value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access Time</td>
<td>Length of time in days that the file has not been accessed</td>
<td>15 days</td>
</tr>
<tr>
<td>Case Sensitive</td>
<td>Defines if case-sensitive (NFS environments) or case-insensitive (CIFS environments) string comparisons will be used.</td>
<td>Off</td>
</tr>
<tr>
<td>Modification Time</td>
<td>Length of time in days that the file has not been modified</td>
<td>15 days</td>
</tr>
<tr>
<td>Minimum Size</td>
<td>Files less than this size will not be deduplicated</td>
<td>24 KB</td>
</tr>
<tr>
<td>Maximum Size</td>
<td>Files greater than this size will not be deduplicated</td>
<td>8 TB</td>
</tr>
<tr>
<td>File Extension Exclude List</td>
<td>Files with the specified extensions will not be deduplicated</td>
<td>None</td>
</tr>
<tr>
<td>Minimum Scan Interval</td>
<td>Frequency with which the deduplication policy engine will scan a deduplication-enabled file system</td>
<td>7 days</td>
</tr>
<tr>
<td>SavVol High Water Mark</td>
<td>Usage capacity percentage of the file system's SavVol at which the space reduction process will not proceed</td>
<td>90%</td>
</tr>
<tr>
<td>Pathname Exclude List</td>
<td>Directories with the specified pathname will not be Deduplicated (File System level setting)</td>
<td>None</td>
</tr>
</tbody>
</table>

Before a deduplication and compression scan begins, the date of the last deduplication scan is checked against the **Minimum Scan Interval**. By default, a deduplication scan will not start if the previous scan ran within the last 7 days. If 7 days have passed, the usage of the SavVol is then checked to verify it is below the **SavVol High Water Mark**. If the SavVol usage is above the **SavVol High Water Mark** either before deduplication is run or while deduplication is running, deduplication is not run or is paused respectively. The SavVol size is also checked because deduplication may cause large changes on the File System in which checkpoints are created.

Next, the File System begins checking for files that do not reside in excluded paths. The user is able to specify which pathnames they wish to be excluded during the scan. Next, a file found is checked to see if the file extension is on the excluded list. The user can update the file extension exclude list to specify different file extensions (such as .txt or .zip) they wish for deduplication to pass over.
If the file is not in an excluded path and does not have an excluded file extension, the file is checked for the last time it was accessed and modified. These times are compared to the current settings, and if the file’s last access and modified dates are older than the values currently set, the next policies are checked for compliance. If the user wishes for a file to be skipped if it was accessed or modified within the last 30 days, this can be set and these files will be skipped.

The size of the file is also checked to see if the policy criterion passes. If the file is larger than the Minimum Size setting and smaller than the Maximum Size setting, the policy check moves on to the other settings. By default, the minimum file size is set to 24 KB and the maximum file size is set at 8 TBs. If the file is too small or too large, deduplication does not run.

Table 5 shows other settings associated with deduplication and compression. Both of these values are user-customizable.

**Table 5 Other deduplication and compression settings**

<table>
<thead>
<tr>
<th>Setting</th>
<th>Definition</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU % High Water Mark</td>
<td>If the CPU reaches this level, deduplication will throttle down (Data Mover level setting)</td>
<td>75%</td>
</tr>
<tr>
<td>CPU % Low Water Mark</td>
<td>If deduplication is throttled down and the CPU level returns to this level, deduplication will throttle up (Data Mover level setting)</td>
<td>40%</td>
</tr>
</tbody>
</table>

The last setting is the Backup Data High Water Mark, which is shown in Table 6. The user can change this value and customize it for their environment.

**Table 6 NDMP backup related setting**

<table>
<thead>
<tr>
<th>Setting</th>
<th>Definition</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Backup Data High Water Mark</td>
<td>Percentage of the logical size of the file that the space-reduced size should be for NDMP to back up the file in its space-reduced format</td>
<td>90%</td>
</tr>
</tbody>
</table>

**Duplicate Detection Method**

When enabling deduplication on VNX File, you have the choice of which Duplicate Detection Method to use. Figure 38 below shows the available choices for the Duplicate Detection Method. The choices are sha1, byte, or off.
Choosing a Duplicate Detection Method

As mentioned previously, File-level deduplication removes duplicate file contents using Single-Instance Storage. This means that duplicate file contents will be identified and single instanced as to reduce space. Only the contents of a file are single instanced, and the existing filenames, security attributes, timestamps, and metadata of the files are retained. Each file that has been single instanced will now point to the location of the single copy of data for those files.

The sha1 algorithm, which is the default deduplication setting, accepts a stream of data less than $2^{64}$ bits in length and produces a 160 bit hash, which is designed to be a unique representation of the data contained within the file. When looking for files that can be deduplicated, the SHA-1 hashes are compared and matching files are identified. After matches are found, the data is deduplicated. The likelihood that two files with different contents are given the same SHA-1 hash is very low, with the odds being less than 1 in $2^{60}$.

Choosing byte will force the system to use a byte-by-byte comparison method when confirming file contents are exact before they are deduplicated. The byte-by-byte comparison occurs only after SHA-1 is used to identify potential matches, which helps to reduce the amount of overhead using byte causes. A considerable amount of overhead can be expected when using byte, especially for large files. Using a byte-by-byte comparison method may be preferred if you need to guarantee the files being deduplicated are exactly the same.

Choosing off will cause Deduplication to be disabled for the File System. If the global Deduplication setting for the File System is enabled, and the Duplicate Detection Method is off, Compression will be the only method of capacity optimization that the File System uses.

Changing the Duplicate Detection Method

If you choose to change the Duplicate Detection Method, the new setting will take effect from that point of change and forward. All files that were deduplicated using a different method previously will remain unchanged. Changing the Duplicate Detection Method is not suggested after enabling deduplication on a File System.

Compression Method

VNX File Compression has two different methods in which space savings can be achieved by using compression: Fast or Deep compression. Fast is the default setting, which optimizes for space savings and speed. Deep is optimized for space savings.
Choosing a Compression Method

There are several key considerations when choosing between the Fast or Deep compression methods. The first consideration is performance. The Deep compression method takes a longer time on each file, compared to Fast, to achieve the highest level of space savings during the compression phase of the operation. Decompression speed is another consideration. As files that are compressed are accessed, they must be decompressed into memory before the I/O is completed. Accessing a file that has been compressed using the Deep method may take more time than if it was compressed using the Fast method. The amount of time depends on the amount of data to rehydrate and the amount of data being accessed.

Another consideration is space savings. Using the Deep method of compression will achieve up to 30% more space savings than if Fast is used. For example, if the Fast method is run on a file and the space savings is 50%, then if the Deep method is used, the space savings on the same file would be near 65%. If achieving the highest space savings is the goal, then the Deep compression method should be used.

The last consideration is compatibility. Not all replication software supports the Deep compression setting. The typical error seen when the replication software doesn’t support the Deep method and attempts a read operation on a file that was compressed using the Deep method is an I/O error. The replication software needs to be checked to see whether Deep compression is supported before it is used.

CIFS Compression Enabled

Table 7 shows an option for controlling whether compression is enabled on CIFS. By default, CIFS compression is enabled. CIFS compression will only occur if Deduplication is turned on or suspended at the File System level. If Deduplication is disabled at the File System level or is in the process of disabling, even if the CIFS Compression Enabled box is checked, CIFS compression will not run.

Table 7 Compression related settings

<table>
<thead>
<tr>
<th>Setting</th>
<th>Definition</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CIFS Compression Enabled</td>
<td>Enables CIFS compression</td>
<td>On</td>
</tr>
</tbody>
</table>

Management

VNX File Deduplication and Compression is enabled on a file system basis. Disabled by default, it can be enabled at the time the file system is created, or after if the user chooses. A single setting to enable File Deduplication and Compression exists, and it is called Deduplication. In VNX OE for File 8.1.8.119, Deduplication has been renamed File-Level Deduplication Enabled with the introduction of VNX Block Deduplication for VNX File. File Deduplication and Compression both work to remove duplicate data and save space.
Figure 39 below shows the **File-Level Deduplication Enabled** setting, which can be enabled at the time the File System is created.

If the user wishes to enable the Deduplication setting on an existing File System, the user first must open the properties window of the File System in question using Unisphere. Figure 40 shows the **Deduplication** setting in the **File System** tab of an existing File System.

**Data Mover level deduplication and compression settings**

On the **Storage** tab in Unisphere, locate the **Deduplication Settings** option under File Storage in the task pane on the right of Unisphere. Figure 41 below shows the
location of this option. **Figure 42** is an example of the **Deduplication Settings** window for the Data Mover.

![Figure 41. Deduplication Settings location](image)

**Figure 42. The Data Mover Deduplication Settings Window**

**File System level deduplication and compression settings**
In the File System window under the **Storage** tab in Unisphere, right-click the file system you wish to view/customize the deduplication and compression settings for, and click Properties. The deduplication and compression settings are found in the **Deduplication Settings** tab of the file system properties window. **Figure 43** is an example of the **Deduplication Settings** tab within the File System Properties window.
Viewing Deduplication Statistics

The user is able to view the results of the deduplication process on the file system data by using the file system's properties page. The deduplication savings is the total amount of space saved for deduplication and compression if both are enabled on the file system. If File Deduplication is disabled, the savings are reported for compression only. The following statistics are reported:

- **Timestamp** of the last successful scan of the file system.
- **Files scanned** — Total number of files that the deduplication policy engine looked at when it last scanned the file system.
- **Files deduped** — Number of files that the deduplication policy engine processed to save space. It also shows the percentage of deduplicated files versus scanned files.
- **Original data size** — Space required to store the data in the file system if it is not deduplicated. This number might exceed the capacity of the file system, in which case the file system is said to be overprovisioned. This is shown by the ratio of the original data size to the file system capacity, which is also displayed.
- **Space saved** — Amount and percentage of space saved by deduplication. This is calculated by subtracting the actual space used to store data after deduplication from the original data size.

After the first scan, statistics are reported as static values based on the last successful scan. **Figure 44** below is an example of the **Deduplication Status** and **Statistics** for a file system.

![Deduplication Savings Example](image)

**VNX File Deduplication and Compression commands**

As with other Unisphere functions, the settings for VNX File Deduplication can also be viewed or changed by using the VNX File Command Line Interface. The command used to view and edit many of the settings is the `fs_dedupe` command. For a detailed synopsis of this command and others, please consult the **VNX Command Line Interface Reference for File** documentation, located on EMC Online Support.

**Conclusion**

VNX storage systems provide powerful capacity efficiency features that can improve effective capacity utilizations. These capacity-optimization features are included with the VNX Operating Environment at no additional cost. The deduplication and compression features for file and block storage offer complementary capacity efficiency opportunities for all data types in the primary storage systems.

VNX Block Deduplication is another way space savings can be achieved. Data contained on deduplicated LUNs within a pool is compared at the 8 KB block level and duplicates are eliminated to save space. Unlike with VNX Block Compression, deduplication is suggested for use on active datasets where a large amount of duplicates exist. EMC suggests using deduplication with Multicore FAST Cache and FAST VP, to achieve better efficiency of these features.

VNX Block Compression runs at the sub-LUN level in 64 KB increments. If at least 8 KB can be saved on the 64 KB chunk, the data is compressed. EMC suggests using VNX Block Compression on static datasets like archives. The best use cases are those where data needs to be stored most efficiently, with a high degree of availability, and where performance overhead can be tolerated.

VNX File Deduplication and Compression work together in providing space savings on VNX File data. While the user has the choice of running compression with or without
deduplication enabled, running them together will help achieve the largest amount of space savings. Deduplication first single instances duplicate files on the File System and then compression tries to reduce the size of the remaining files. The savings achieved while using VNX File Deduplication and Compression will vary as the savings entirely depend on the files being stored on the file system.

References
The following white papers are available on EMC Online Support:

- Virtual Provisioning For The VNX2 Series – Applied Technology
- EMC VNX2 Multicore FAST Cache
- EMC VNX2 FAST VP – A Detailed Review
Appendix A: NTFS File System Conditioning with SDelete

Many file systems do not efficiently reuse the space associated with deleted files. When files are deleted from NTFS file systems, the deleted files’ data continues to be stored in the file system until it is overwritten by new data. When files are frequently deleted, the free space in the file system may gradually become filled with deleted file data that is no longer accessible by the file system. Deleted file data reduces the effectiveness of EMC Data Compression. The retention of deleted file data is a characteristic of the file system and is relevant regardless of whether LUNs are presented directly to a Windows server or the Windows server resides in a VM.

Data Compression processes blocks associated with deleted files the same way that data is processed for valid files. Unused (never-used) blocks in the NTFS file system compress to zero consumed capacity, but deleted file blocks are only as compressible as the deleted file data that continues to be stored on them.

The SDelete utility from Microsoft replaces deleted file blocks with zeros when it is invoked with the -c option. Note that deleted file blocks cannot be removed by defragmentation or reformatting the file system. The blocks zeroed by SDelete do not consume any capacity once they are processed by Data Compression. This can have a profound impact on compression results.

For example, let’s assume a 100 GB file system has files that are 1.5:1 compressible (33 percent space savings). This file system resides on a 100 GB Classic LUN. Also assume that the file system reports used space to be 60 GB, but since the file system has been in use for some time, there are actually 90 GB of used blocks in the file system. This means that only 10 GB of the file system capacity is unallocated to data.

Without SDelete, the resulting compressed LUN will consume roughly 60 GB in the pool, since all 90 GB worth of blocks is compressible at a 1.5:1 ratio. Thin LUN metadata may add another 3-4 GB of consumed capacity. The 10 GB of unallocated data does not consume any capacity once it is compressed. This results in overall capacity savings of roughly 40 GB.

If SDelete runs on the file system prior to being compressed, Data Compression is more effective. There is still 60 GB of data, but the extra 30 GB of deleted file blocks and the 10 GB of unallocated capacity are overwritten by zeros by SDelete. In this case, the compressed LUN only consumes 40 GB in the pool for a total savings of 60 GB. The data itself compressed at a ratio of 1.5:1, but the zeroed capacity does not consume any space in the compressed LUN. In this example, using SDelete yielded an additional 50 percent of space savings.

SDelete writes to all space not consumed by a valid file data. If the file system resides on a Thin LUN, SDelete causes the Thin LUN to become fully consumed. Therefore, users must be sure they have adequate pool capacity if they choose to run SDelete on a Thin LUN. On Classic LUNs or Thick LUNs hosting NTFS file systems, run SDelete prior to enabling compression to maximize space savings. Additional information, including the download of the utility, can be found at Microsoft’s TechNet webpage: http://technet.microsoft.com.
Appendix B: File Data Migration to VNX with Deduplication and Compression Example

The following examples identify a customer wanting to migrate data from a Windows server to the VNX system. These real world scenarios show what happens when you migrate data and the importance of deduplication being enabled prior to migration.

Note: The fsUtils package contains the following utilities:
- fsScan: Produces data files that act as small databases of all the metadata in a file system or group of file systems.
- exScan: Provide assessment information for the current usage breakdown of existing Exchange mail characteristics, such as size and last access time, and for the assessment of potential space savings from an archiving solution implementation.
- fsReports: Create reports from fsScan data files.
- fsDiff: Compare two fsScan output files

In these examples, we gathered the metadata with the fsscan utility, which creates a .dtl file. Then, we ran reports on the dtl file with the fsreports utility. This gave us information about files to deduplicate and archive.

**Customer A**
Customer has a Windows server that will be migrated to VNX:
- 2TB of MS Office data
- Highly visible project
- Existing performance issues

<table>
<thead>
<tr>
<th></th>
<th>File System</th>
<th>Deduplication Results</th>
<th>Savings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Analysis of Windows server</td>
<td>2TB</td>
<td>Not enabled</td>
<td>0</td>
</tr>
<tr>
<td>FsUtil output</td>
<td>2,471GB</td>
<td>4,056,632 files can be archived</td>
<td>73% of all files have gone 180+ days untouched</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3,522,207 files can be deduplicated</td>
<td>2,236GB combined size, or 90% of total size at 40% compression: 893GB potential savings</td>
</tr>
<tr>
<td>After Migration to VNX</td>
<td>1,590GB</td>
<td>3,461,285 files deduplicated</td>
<td>881GB space saved</td>
</tr>
</tbody>
</table>

**Customer B**
Customer has an older NAS that needs to be migrated to a VNX:
- Host based migration
- Highly visible
- Must have replication set from day one

Deduplication was enabled after IP Replication & Checkpoints & post-migration

<table>
<thead>
<tr>
<th></th>
<th>File System</th>
<th>Deduplication Results</th>
<th>Savings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Analysis</td>
<td>760GB</td>
<td>Not enabled</td>
<td>0</td>
</tr>
</tbody>
</table>
### File System Deduplication Results

<table>
<thead>
<tr>
<th>of Older NAS</th>
<th>File System</th>
<th>Deduplication Results</th>
<th>Savings</th>
</tr>
</thead>
<tbody>
<tr>
<td>FsUtil</td>
<td>760GB</td>
<td>770,725 files can be archived</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>844,761 files can be deduplicated</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>69% of all files have gone 180+ days untouched</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>718GB combined size, or 75% of total size at 40% compression: 288GB potential savings</td>
<td></td>
</tr>
<tr>
<td>After Migration to VNX</td>
<td>?? ??</td>
<td>44,862 files deduplicated</td>
<td>9GB space saved</td>
</tr>
</tbody>
</table>

**WHAT JUST HAPPENED??**
- SavVol was at 82% of 137GB size (Checkpoints & Replication overhead)
- Deduplication had only 8% of 137GB, or 10GB to operate
- Once the FS had 10GB of changes, deduplication stopped for a week

After extending the SavVol by 300GB:

<table>
<thead>
<tr>
<th>File System Deduplication Results</th>
<th>Savings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Analysis of Older NAS</td>
<td>760GB</td>
</tr>
<tr>
<td>FsUtil output</td>
<td>760GB</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>After Migration to VNX</td>
<td>276GB</td>
</tr>
</tbody>
</table>

**Summary:** Do not turn on checkpoints before doing migration with deduplication and compression enabled.

When you turn on checkpoints before doing a migration with deduplication enabled, the system will deduplicate files to clear space, freeing up blocks to be used. This could cause the checkpoint SavVol to grow very large as writes will start to fill up those freed up blocks. This will cause the SavVol to track those new blocks as needing to be saved into the SavVol. This can cause the SavVol to keep expanding to track all the new changes.
## Appendix C: Comparing VNX Block Deduplication for VNX File Data and VNX File Deduplication and Compression

When determining which storage efficiency feature to use for VNX File data, many considerations need to be investigated. The chart below outlines the major differences between the two deduplication methods for VNX File Data.

**Granularity** – The method of deduplication used by the feature  
**Data Selection** – Method in which data is considered for deduplication  
**Realized Savings** – The level in which space savings is returned to  
**Space Savings Mechanism** – The software feature used to achieve space savings  
**Supported LUN Type** – The supported LUN types as the underlying storage

<table>
<thead>
<tr>
<th>Consideration</th>
<th>VNX Block Deduplication</th>
<th>VNX File Deduplication</th>
</tr>
</thead>
<tbody>
<tr>
<td>Granularity</td>
<td>8 KB Fixed Block (Within the Deduplication Container of a Block Storage Pool)</td>
<td>Single Instance Storage of Files (Within a single File System)</td>
</tr>
<tr>
<td>Data Selection</td>
<td>All data within the Deduplication Container</td>
<td>Policy based deduplication</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Filtering Options Include: file modified time</td>
</tr>
<tr>
<td></td>
<td></td>
<td>last accessed time</td>
</tr>
<tr>
<td></td>
<td></td>
<td>size and extension</td>
</tr>
<tr>
<td>Realized Savings</td>
<td>Block LUNs or Block Storage Pool shows reduced used capacity</td>
<td>File System used capacity reduced</td>
</tr>
<tr>
<td>Space Savings Mechanism</td>
<td>Fixed-Block Deduplication</td>
<td>File Deduplication and Compression</td>
</tr>
<tr>
<td>Supported LUN Type</td>
<td>Deduplication enabled Thin LUNs</td>
<td>Classic LUNs, Thick or Thin Storage Pool based LUNs</td>
</tr>
</tbody>
</table>

Note: For more on VNX Block Deduplication and VNX Block Deduplication for VNX File, please review the appropriate sections found within this document.