VERITAS

Veritas NetBackup Flex Appliances Best Practices

This document provides best practice recommendations to achieve optimized performance for backup, restore, duplication and replication workloads on NetBackup™ Flex Appliances.

White Paper | November 2021

Contents

Executive Summary . . . 3

Sizing Flex Container Instances . . . 3

- Primary Server Instances . . . 3
- MSDP and WORM Server Instances . . . 3
  - Pitfalls . . . 4
- NetBackup Appliance to Flex Consolidation Considerations . . . 4
  - Pitfalls . . . 5

Flex 5250 Appliance Cloud Tiering Sizing . . . 5

- Recommended Configuration of a Flex 5250 Appliance with MSDP-C and No External Storage . . . 6
- Guardrails . . . 6

Flex Appliance Tuning . . . 6

- Default MSDP Setting . . . 6
- Media MSDP Instance Memory Tuning . . . 7
  - MaxCacheSize . . . 7
  - MaxCacheSize Tuning Examples . . . 7
  - MSDP Deduplication Multi-Threaded Agent Tuning . . . 8
  - LUN Creation Sequence . . . 9
  - Best Practices for LUN Sharing . . . 10
  - Multiple LUNs for One Media MSDP Container . . . 11

Tuning MSDP-Direct Cloud Tier . . . 12

- Considerations when More Than One Media Instance Has at least One Cloud LSU Configured . . . 13

Appendix . . . 15

- Flex Tuning Parameters . . . 15
  - Default Tunings in Flex 1.2 . . . 15
  - Default Tunings in Flex 1.3 . . . 15
  - Default Tunings in Flex 2.0 and Flex 2.0.1 . . . 16
- Manual Tunings Needed in Flex 1.2 . . . 16
- Manual Tunings Needed in Flex 1.2 and Flex 1.3 . . . 17
- Manual Tunings Needed in Flex 2.0 . . . 17

Versions . . . 19

Executive Summary

Veritas NetBackup Flex Appliances allow you to configure multiple NetBackup media, MSDP-Cloud and Primary Server instances on a single hardware platform. Each NetBackup instance is running in a container and shares the hardware resources such as CPU, memory, I/O and network.

Proper sizing and tuning are critical to optimize Flex Appliance performance. This paper will cover the sizing guidelines and customized performance tuning for Flex Appliances.

Sizing Flex Container Instances

When determining the number of Flex container instances to deploy per Flex node, it is important not to conflate the maximum number of instances supported as a recommendation with the number of instances that should be deployed. No two instances are the same. The key factors to consider when determining the ideal number of instances to deploy per Flex node include:

- Workload types and sizes
- Data characteristics (third-party encryption, millions of small files)
- Backup methodology and retentions
- Data lifecycle and secondary operations (optimized duplications, auto image replication [AIR] or tape-out)
- Backup windows
- Recovery time objective (RTO), recovery point objective (RPO) and service-level agreements (SLAs)
- Finite system resources (CPU, memory, network)
- Storage (the number of physical disks and associated LUNs)

Primary Server Instances

A NetBackup Primary Server instance tends to be I/O-intensive, albeit small in capacity compared to an MSDP or WORM Server instance. When mapping out the sizing of the Primary Server instance, consider that storage LUNs for Flex 5350 are 40 TB when the disks are 4 TB in size, and 80 TB when the disks are 8 TB in size. Because a Primary Server instance doesn't require an entire LUN, part of that LUN will likely be shared with another Flex instance. However, it is important not to share a single LUN between more than two Flex instances. Ideally, the goal is to not share any LUNs between instances.

MSDP and WORM Server Instances

When sizing MSDP or WORM Server instances, determine the workload types and front-end terabytes (FETB) you want to protect. Review the number of LUNs available and plan out the LUNs that will be allocated to each instance. As part of the planning phase, ensure the storage allocated can accommodate growth as well as a few days of storage lifecycle policy (SLP) backlogs for secondary operations. Maintenance will be required periodically, so ensure the solution can accommodate such scenarios.

Grouping workload types together optimizes dedupe performance as well as simplifying the solution to make predictive analysis and trending easier. When sizing the instances, also factor in a conservative dedupe rate like 80 percent. Some workloads may only realize a dedupe rate of 65 percent, and others will realize dedupe rates as high as 90 percent. The dedupe rate depends on various factors, including the rate of change and whether the workload—and its data characteristics—lends itself to good dedupe performance.

Secondary operations like duplication, replication via AIR and duplication to tape use memory, CPU and I/O resources. Duplications to tape are particularly resource-intensive because images on dedupe storage must be rehydrated as part of the duplication process to tape. This process results in intensive read and write operations. Additionally, Flex instances that have both Advanced Disk and MSDP pools will generate intensive read and write operations when specific workloads like database transaction logs are written to an Advanced Disk pool prior to being duplicated to an MSDP pool. These types of intensive read and write operations will use significant I/O and CPU resources.

MSDP is memory intensive. To ensure MSDP instances don't run into a memory contention issue, plan for the MSDP cache to use 1 GB of memory for every 1 TB of disk storage. For example, an MSDP pool that is 480 TB in size should have at least 480 GB of memory resources to allow for optimal MSDP cache operations. Flex instances that are consistently busy with backup and secondary operations should also have enough memory above the MSDP cache requirements to accommodate the resources required to process the workloads.

Pitfalls

Oversubscribing the number of Flex instances that run on a single Flex node is a typical pitfall in business environments.

Typically, I/O contention is the number one issue due to too many instances sharing the same LUNs. For example, an organization that has built three Flex instances—one Primary instance, one MSDP instance and one WORM instance—to run on one Flex node with one storage shelf of 8-TB drives, resulting in six 80-TB LUNs, will often start to experience I/O contention as soon as the third instance is active. To address this issue, the organization requires additional storage to ensure all three instances have no shared LUNs. The solution also required some tuning as well as workload balancing to achieve optimal performance. This scenario shows a clear correlation between I/O contention and instance oversubscription when the storage and associated I/O resources aren't sized properly.

Another oversubscription scenario can arise when a solution is designed for capacity, but not sized for compute resources such as memory and CPU. For example, an organization t has a solution consisting of four storage shelves containing 8-TB drives and one Flex 53xx node with 1.5 TB of memory. In this scenario, there isn't enough memory to support the use of all the storage on a single node. Two instances with 960-TB MSDP pools each or four instances with 480-TB MSDP pools each isn't sized to handle the compute resource requirements. Another Flex 53xx node configured as a Flex HA would be the appropriate choice to support 1.8 PB of storage and provide the organization some additional resiliency. Of course, following best practices around tuning and LUN allocation is always required.

NetBackup Appliance to Flex Consolidation Considerations

NetBackup Appliance to Flex consolidation initiatives are becoming more common as companies realize the benefits of the Flex platform. As they plan for such consolidation efforts as part of a hardware refresh, correct sizing of the target Flex environment is paramount to support existing and future workloads successfully.

Unlike a typical sizing effort for a new Flex environment, an existing NetBackup Appliance footprint that will be consolidated with new hardware on the Flex platform requires additional information and planning. Because there are existing workloads that are being processed by the legacy NetBackup Appliance solution, it is critical to gather the following key information about the existing environment prior to sizing the new target Flex environment.

- The number of NetBackup Appliances being consolidated, including:
  - Model
  - Disk pool size(s) – MSDP and/or Advanced Disk
  - % of capacity in use per disk pool
  - Memory in GB
  - Type and number of CPU cores
  - Network info (speed and number of interfaces)
- Workload types and FETB
- Current dedupe rates
- Average rate of change per day
- Expected growth per year
- Current backup window
- Current SLAs, RPO and RTO
- All secondary operations (optimized duplications, AIR, duplication to tape
- Special data characteristics (third-party encryption, millions of little files)

We don't recommend sizing any environment just for capacity, especially in this case. The sheer number of physical disks that comprise all the MSDP and Advanced Disk pools in the legacy environment drive a significant piece of the sizing criteria for the new Flex environment. The number of physical disks, also referred to as spindles, in the legacy NetBackup Appliance environment will likely exceed the number of physical disks in a consolidated Flex environment when sized only by capacity. If an organization is handling the backups and secondary operations of a certain number of clients and FETB within a certain period of time, albeit distributed across a large pool of appliances, the cumulative CPU, memory and I/O bandwidth is likely to be considerable.

One way to help ensure the new Flex environment has the necessary I/O bandwidth is to consider 4-TB drives. Although the use of 4 TB increases the physical footprint when compared to 8-TB drives, it provides significantly more I/O bandwidth because it doubles the number of spindles.

Based on past sizing initiatives, a general rule of thumb would be that a single Flex node can consolidate two to three legacy NetBackup Appliances provided there is enough storage capacity, I/O bandwidth and memory to accommodate the workloads being migrated. Please keep in mind this is a general rule of thumb. When consolidating large NetBackup Appliances to Flex, there are some scenarios when a single Flex node may only consolidate one legacy NetBackup Appliance. Usually, a legacy NetBackup Appliance environment is a mix of older NetBackup 52xx and 53xx Appliances.

Although the use of newer NetBackup features like VMware Protection Plans, Accelerator and MSDP-C can help make the processing of existing workloads more efficient and faster, the amount of CPU, memory and I/O bandwidth is finite. We must consider how long it takes for current workloads to back up and complete secondary operations prior to sizing the target Flex environment.

Pitfalls

An organization with 50+ legacy NetBackup Appliances initiated a consolidation and hardware refresh in a single project. It was eager to rush past the data gathering and analysis phase of the initiative and simply size based on capacity. Although the organization could have gathered and analyzed information about the legacy NetBackup Appliances, it insisted it didn't have time to do the necessary sizing based on more than capacity. Instead, it ran a nbdeployutil on each Primary Server to gather only workload and capacity information. The end result was a solution consisting of three Flex HA pairs.

It is also important to note that the organization didn't know much about how the NetBackup Flex platform differed from traditional NetBackup Appliances. This lack of knowledge resulted in an initial deployment that significantly diverged from Flex best practices. Even after Veritas assisted the organization with tuning and optimizing the configuration, only about 25 percent of the workloads were migrated successfully. It was only after the organization purchased and deployed another six Flex HA pairs that it was able to migrate the rest of its workloads successfully. The organization clearly learned the hard way that sizing based on capacity alone results in unsuccessful consolidation efforts.

Although the organization initially purchased 8-TB drives with the first three Flex HA pairs, it found the I/O bandwidth wasn't enough to process the backups and secondary operations within the same backup window as in the legacy environment. Additional storage and Flex nodes were required to complete the migration.

Even though this is just one scenario, it illustrates how a large consolidation effort can be set up to fail when an organization doesn't do the necessary data, analysis and planning in advance.

Flex 5250 Appliance Cloud Tiering Sizing

The Flex 5250 Appliance can support a multi-tier NetBackup environment. Due to its limited resources in its smallest configuration, customers must be mindful of the load put on the system. The smallest Flex 5250 configuration consists of internal storage (no external shelves) with 9 TB of usable disk capacity and 64 GB of memory. This configuration can support a single NetBackup Primary Server container and a single NetBackup Media Server container with a standard (not immutable) MSDP pool. This configuration will also support MSDP cloud-tiering, which should not run concurrently with backups.

The backup configuration using server-side deduplication has been tested with up to 90 concurrent backup streams before seeing errors. Maximum throughput was achieved with only 16 concurrent streams, however, so we recommend not exceeding 16 concurrent backup streams using server-side deduplication. Using client-side deduplication reduces the resources required on the appliance and can likely support more concurrent backup streams. Organizations should always use the lowest number of backup streams required to achieve maximum throughput.

Although the minimal Flex 5250 Appliance configuration can support MSDP backups and MSDP cloud tiering on the same appliance, it does not have enough resources to support these processes running concurrently. Instead, cloud-tiering should be done during a different time window than backups. Cloud tiering requires 1 TB of free space per cloud tier target. Each terabyte required for cloud tiering comes from the MSDP pool. Organizations must take this requirement into account and ensure the MSDP space required to support the cloud tier(s) is always available. When cloud tiering is in use, change the MaxCacheSize NetBackup parameter to 30 percent; otherwise, leave it at its default value.

Recommended Configuration of a Flex 5250 Appliance with MSDP-C and No External Storage

- 36 TB capacity
- 64 GB memory
- One Primary Server
- One Media Server
- Standard MSDP pool (not immutable)
- Cloud tier to S3 target(s)

Guardrails

Concurrent streams: 16 for peak throughput, tested up to 90 without job errors

- Client-side dedupe allows more concurrent streams
- 1 TB free space required per cloud tier
- MaxFileSize (MSDP data container size): 64 MB (default value)
- MaxCacheSize: 30 percent (only change if using cloud-tiering)

Flex Appliance Tuning

MaxCacheSize is an MSDP parameter that controls MSDP instance fingerprint cache size to limit the maximum amount of memory a media instance can use for fingerprint caching and reserves enough RAM for the operating system and application processes. Without this limitation, MSDP can consume an excessive amount of RAM for fingerprint caching and cause memory starvation for other processes running on the system. Excessive swapping can result and slow down overall system performance.

Default MSDP Setting

Because all instances share hardware resources on Flex Appliances, proper tunings can help optimize performance and reduce support cases.

Before we discuss how to customize MaxCacheSize, Table 1 shows the default manufacture MaxCacheSize setting.

Table 1. MSDP Fingerprint Cache Configuration

Version Date	Before NetBackup 8.2	NetBackup 8.2 and later
Each Media MSDP container fingerprint cache size (MaxCacheSize)	10% of system memory	50% of system memory .

For example, the Flex 5340 Appliance is shipped with 768 GB of memory by default. If you apply the memory upgrade kit that doubles the physical memory to 1.5 TB, you can relax the 50 percent limitation to 70–75 percent if the appliance is not overloaded with a large number of concurrent jobs.

Media MSDP Instance Memory Tuning

The MaxCacheSize setting determines how many fingerprint indexes can be cached in memory per instance, which can potentially influence the deduplication ratio. Any generated index would be identified as a new index if there is no matching index found in the cache. When the system lacks cache space, the least recently used index will be deleted for the new index. When MaxCacheSize is set too low, this scenario will happen more often. Fingerprint cache miss increases the write I/O to the storage pool, decreases the deduplication ratio and slows down the overall backup performance.

MaxCacheSize

The following example shows how to calculate MaxCacheSize based on storage size.

> The total amount of RAM required for a media MSDP instance on a Flex Appliance is based on "1 GB of RAM for each TB of storage." We recommend using no more than 50 percent of the RAM for fingerprint caching.

Example:

If the storage allocated for a media/MSDP instance is 80 TB, the total RAM required to run the instance would be 80 GB. Of the 80 GB RAM, we recommend using no more than 50 percent for fingerprint caching. The fingerprint cache should be set at 40 GB RAM. A Flex 53xx Appliance by default is configured with 768 GB of RAM and 40 GB is roughly 5 percent of the 768 GB system RAM. Therefore, the MaxCacheSize should be set to 5 percent. The 5 percent is derived and rounded based on the formula ((40GB/768GB) * 100) %.

Note: The 50 percent MaxCacheSize setting is a best practice guide, not a firm requirement. The ratio will hold only if the size of RAM on the appliance is exactly 1vGB of RAM per TB of storage. In real customer deployments, the RAM size could be either greater or less than 1 GB per TB of storage. Therefore, the aggregate MaxCacheSize for all media instances will be either less than or greater than 50 percent, respectively. It is normal for the aggregate MaxCacheSize to be slightly higher than 50 percent of the RAM, especially in a newly deployed Flex Appliance, because the deduplication engine allocates Fingerprint cache as needed. In a fresh deployment, the number of Fingerprints will be small and only a small portion of the MaxCacheSize will be used for caching. As the storage pool is filled up, more RAM will be consumed to cache the increasing Fingerprints up to the MaxCacheSize. When the aggregate MaxCacheSize is much higher than 50 percent, the appliance will start to show signs of memory starvation such as increased swapping activities and a slowdown of overall system performance.

MaxCacheSize Tuning Examples

This section gives two examples of MaxCacheSize configurations for a Flex Appliance with 4-TB and 8-TB storage shelves. Both examples have the RAM size much lower than the recommended 1 GB of RAM per TB of storage, therefore the aggregate MaxCacheSize is approximately 60 percent. For the first example, we recommend adding additional memory. For the second example, we recommend deployed on a Flex HA Appliance and separating the workload into two containers, each running on a separate head node.

Example 1. 4-TB disk drives with 768 GB of RAM

Parameter	1 MSDP	2 MSDPs	4 MSDPs	6 MSDPs
MSDP Pool Size	MSDP 1: 960 TB, 24 LUNs MaxCacheSize calculation: 960*0.5=480GB (480/768GB) * 100 = 62.5	MSDP 1: 480 TB, 12 LUNs MSDP 2: 480 TB, 12 LUNs	MSDP 1: 240 TB, 6 LUNs MSDP 2: 240 TB, 6 LUNs MSDP 3: 240 TB, 6 LUNs MSDP 4: 240 TB, 6 LUNs	MSDP 1: 160 TB, 4 LUNs MSDP 2: 160 TB, 4 LUNs MSDP 3: 160 TB, 4 LUNs MSDP 4: 160 TB, 4 LUNs MSDP 5: 160 TB, 4 LUNs MSDP 6: 160 TB, 4 LUNs
MaxCacheSize per MSDP Instance	60% (rounded from 62.5)	30%	15%	10%

Example 2. 8-TB disk drives with 1.5 TB of RAM

Flex Appliances support 1.9 TB storage, and each media server instance supports no more than 960 TB . You can create two media server instances, each with 960 TB.

Parameter	1 MSDP	2 MSDPs	4 MSDPs	6 MSDPs
MSDP Pool Size	MSDP 1: 1920 TB, 24 LUNs 1920/2=960GB (960/2*768 GB) * 100 = 62.5	MSDP 1: 960 TB, 12 LUNs MSDP 2: 960 TB, 12 LUNs	MSDP 1: 480 TB, 6 LUNs MSDP 2: 480 TB, 6 LUNs MSDP 3: 480 TB, 6 LUNs MSDP 4: 480 TB, 6 LUNs	MSDP 1: 320 TB, 4 LUNs MSDP 2: 320 TB, 4 LUNs MSDP 3: 320 TB, 4 LUNs MSDP 4: 320 TB, 4 LUNs MSDP 5: 320 TB, 4 LUNs MSDP 6: 320 TB, 4 LUNs
MaxCacheSize per MSDP Instance	60% (rounded from 62.5)	30%	15%	10%

MSDP Deduplication Multi-Threaded Agent Tuning

The Deduplication Multi-Threaded Agent (mtstrmd) is designed to improve MSDP backup performance for systems with a low number of concurrent jobs. Each media server instance has its own mtstrmd that is enabled by default. The default mtstrmd session number is automatically calculated at the time of installation and is captured in the mtstrmd.conf, which is under the parameter MaxConcurrentSessions. The mtstrmd processes are CPU intensive and too many mtstrmd sessions can cause a CPU bottleneck and degrade the overall backup performance.

The default session number works fine for a single media instance on a Flex Appliance. However, when deploying multiple media instances on a single Flex Appliance head node, you should reduce the MaxConcurrentSessions for each instance. The aggregate number of sessions should not exceed the MaxConcurrentSessions listed in Table 2.

Table 2. Aggregate MaxConcurrentSessions Limits

Flex Appliance Model	MaxConcurrentSessions
Flex 5150	3
Flex 5250	9
Flex 5340	15
Flex 5350	19

For example, if you deploy three media server instances with similar workload and performance requirements on a single Flex 5340 Appliance, we recommend you set the MaxConcurrentSessions to 5 for each media instance. If one of the media instances has a high-performance requirement, set the MaxConcurrentSessions to 15 on that media instance and disable the mtstrmd on the other two media server instances.

To change the MaxConcurrentSessions number, log into the media container, edit the value inside /usr/openv/lib/ost-plugins/mtstrm.conf and then kill the mtstrmd process inside the container when there's no job using mtstrmd. It will start automatically when the new job kicks in and the new MaxConcurrentSessions setting will take effect.

Media MSDP Instance Storage Allocation

The storage allocation plays an important role in media MSDP performance. Simply allocating enough storage to meet the size requirement of each instance is not sufficient for optimal performance.

When multiple MSDP containers share the same LUN(s), multiple backup jobs from the containers can write to the same LUN concurrently and result in I/O contention. The I/O contention degrades performance, especially for the I/O-intensive workload.

Avoiding or reducing multiple instances sharing the same LUN is critical to reduce I/O contention and achieve optimal I/O performance.

LUN Creation Sequence

A fully populated Flex 53xx storage shelf such as RBOD/EBOD is configured with 6 x RAID6 LUNs. Each RAID6 LUN is built with 13 disk drives or 11 data disks and 2 parity disks called (11D+2P) LUN. A half-populated shelf is configured with 3 x RAID6 LUNs. Figure 1 displays Flex 53xx storage options.

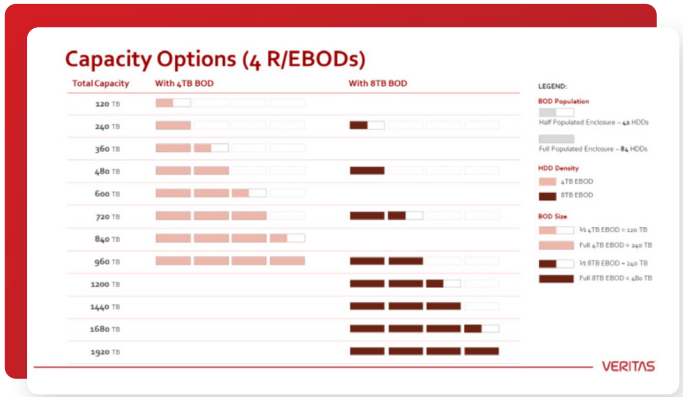

Figure 1. Flex 53xx Appliance storage options.

The correct container creation sequence can prevent multiple media MSDP containers from sharing the same LUN(s).

LUN creation uses the round-robin method. Let's say we want to provision six Primary containers and six media MSDP containers on a single, fully populated storage shelf with 4-TB drives. There are six 40-TB LUNs. Six Primary containers with 5 TB storage each are created first. Using the round-robin method, these containers are created on a different LUN. Then we create six 34-TB media MSDP containers on six different LUNs as shown in the diagram below. The disk I/O is isolated by running NetBackup jobs concurrently on six media MSDP instances on six different LUNs.

```
[root@nbapp842 ~]#  df -h | grep msdpdata
/dev/vx/dsk/vxosdg/vWWXA_msdpdata-0                    34T  9.2T   25T  28% /var/lib/docker-veritas-plugin/vWWXA_msdpdata-0_vxosdg
/dev/vx/dsk/vxosdg/v1JTY_msdpdata-0                    34T   11T   24T  30% /var/lib/docker-veritas-plugin/v1JTY_msdpdata-0_vxosdg
/dev/vx/dsk/vxosdg/vFTpX_msdpdata-0                    34T   10T   24T  30% /var/lib/docker-veritas-plugin/vFTpX_msdpdata-0_vxosdg
/dev/vx/dsk/vxosdg/vSzVE_msdpdata-0                    34T   11T   24T  31% /var/lib/docker-veritas-plugin/vSzVE_msdpdata-0_vxosdg
/dev/vx/dsk/vxosdg/vMoeR_msdpdata-0                    34T  9.2T   25T  28% /var/lib/docker-veritas-plugin/vMoeR_msdpdata-0_vxosdg
/dev/vx/dsk/vxosdg/vA8E1_msdpdata-0                    34T   11T   24T  31% /var/lib/docker-veritas-plugin/vA8E1_msdpdata-0_vxosdg
[root@nbapp842 ~]#  df -h | grep catalog
/dev/vx/dsk/vxosdg/vXoMo_catalog                      5.0T  3.2G  5.0T   1% /var/lib/docker-veritas-plugin/vXoMo_catalog_vxosdg
/dev/vx/dsk/vxosdg/vg27z_catalog                      5.0T  3.2G  5.0T   1% /var/lib/docker-veritas-plugin/vg27z_catalog_vxosdg
/dev/vx/dsk/vxosdg/vFUOz_catalog                      5.0T  3.3G  5.0T   1% /var/lib/docker-veritas-plugin/vFUOz_catalog_vxosdg
/dev/vx/dsk/vxosdg/vVDj3_catalog                      5.0T  3.2G  5.0T   1% /var/lib/docker-veritas-plugin/vVDj3_catalog_vxosdg
/dev/vx/dsk/vxosdg/vgjQa_catalog                      5.0T  3.3G  5.0T   1% /var/lib/docker-veritas-plugin/vgjQa_catalog_vxosdg
/dev/vx/dsk/vxosdg/vzhnu_catalog                      5.0T  3.2G  5.0T   1% /var/lib/docker-veritas-plugin/vzhnu_catalog_vxosdg
```

```
[root@nbapp842 ~]#  vxprint -ht | egrep "sd" | egrep -v "^v|^pl"| egrep "catalog"
sd F000285A3EB979055B01000000-01 vFUOz_catalog-01 F000285A3EB979055B01000000 0 10737418240 0 vrts_0_4_data ENA
sd F00028E783CF79055B01000000-01 vVDj3_catalog-01 F00028E783CF79055B01000000 0 10737418240 0 vrts_0_5_data ENA
sd F000285A3E9379055B01000000-01 vXoMo_catalog-01 F000285A3E9379055B01000000 0 10737418240 0 vrts_0_2_data ENA
sd F000285A3EE179055B01000000-01 vgjQa_catalog-01 F000285A3EE179055B01000000 0 10737418240 0 vrts_0_6_data ENA
sd F00028E783A579055B01000000-01 vg27z_catalog-01 F00028E783A579055B01000000 0 10737418240 0 vrts_0_3_data ENA
sd F00028E783F279055B01000000-01 vzhnu_catalog-01 F00028E783F279055B01000000 0 10737418240 0 vrts_0_7_data ENA
[root@nbapp842 ~]#  vxprint -ht | egrep "sd" | egrep -v "^v|^pl"| egrep "msdp"
sd F00028E783F279055B01000000-02 vA8E1_msdpdata-0-01 F00028E783F279055B01000000 10737418240 73014444032 0 vrts_0_7_data ENA
sd F000285A3EB979055B01000000-02 vFTpX_msdpdata-0-01 F000285A3EB979055B01000000 10737418240 73014444032 0 vrts_0_4_data ENA
sd F000285A3EE179055B01000000-02 vMoeR_msdpdata-0-01 F000285A3EE179055B01000000 10737418240 73014444032 0 vrts_0_6_data ENA
sd F00028E783CF79055B01000000-02 vSzVE_msdpdata-0-01 F00028E783CF79055B01000000 10737418240 73014444032 0 vrts_0_5_data ENA
sd F000285A3E9379055B01000000-02 vWWXA_msdpdata-0-01 F000285A3E9379055B01000000 10737418240 73014444032 0 vrts_0_2_data ENA
sd F00028E783A579055B01000000-02 v1JTY_msdpdata-0-01 F00028E783A579055B01000000 10737418240 73014444032 0 vrts_0_3_data ENA
```

Best Practices for LUN Sharing

You cannot avoid LUN sharing when the instance's storage requirement is much smaller than the LUN size. Two containers sharing the same LUN is acceptable, although the I/O performance would be degraded when the two containers are running I/O workloads concurrently. Staggering the workload schedules will help mitigate the I/O contention.

The best practice is to have no more than two media MSDP containers share a LUN.
Staggering the workload schedules will help mitigate the I/O contention.

With careful planning to minimize possible I/O contention, it is still possible to achieve good I/O performance even with some LUN sharing. Here are the best practices to follow if you cannot avoid LUN sharing:

1. Limit to no more than two instances sharing the same LUN, if possible.
2. Choose 4-TB or 8-TB shelves based on the MSDP storage pool size profile. Choose 4-TB shelves if the storage pool of multiple instances is 20 TB or less. This option can reduce the need for more than 2 instances sharing the same LUN.
3. Create the instance with the highest I/O performance requirement first. The storage pool of the first created instance will occupy the outer layer of the LUN, which can outperform the storage pool located on the inner layer of the LUN.

4. Use the backup schedule and Storage Lifecycle Policy (SLP) to reduce I/O contention and achieve the best I/O performance.

   - Multiple instances sharing the same LUN will not affect performance unless the instances sharing the LUN are active at the same time. If the workload of small instances can be finished in few hours, then stagger the activation of each instance to avoid job overlap and thus the I/O contention.

5. Instances with a high dedupe ratio do not generate a lot of write I/Os. If multiple instances (two or more) must share the same LUN and you cannot stagger the backup workload, then choose instances with high dedupe ratios to share the LUN. You should stagger the SLP jobs that generate read I/O for each instance, especially when more than two instances share the same LUN.

6. In some cases, the leftover space (we will call it subdisk) from any LUN on the storage shelf may not be enough to meet an instance's storage pool requirement. An alternative is to concatenate two or more subdisks, each from a different LUN, to form the require volume size. In this case, the order in which you choose the subdisks can impact performance. Carefully choosing the subdisks can avoid unnecessary I/O contention. The following example demonstrates how to choose subdisks to avoid I/O contention.

7. Assume LUN 1 has two subdisks, SD1 and SD2 and LUN 2 has two subdisks, SD3 and SD4. The storage pool of instance A requires SD1 and SD3, and instance B ends up with SD2 and SD4. By default, the subdisks are concatenated together to form a larger volume and write I/O will fill up the first subdisk before going to the second subdisk. So if instance A configures SD1 as the first subdisk, then instance B should configure SD4 as the first subdisk. Doing so will ensure that instance A and B will not be writing to the same LUN some of the time, thus avoiding I/O contention.

   - Note: Sometimes you discover LUN sharing after the media/MSDP instances are deployed and backup data already resides on the subdisks. To reduce I/O contention in this case, you may need to change the storage allocation by moving subdisks around. You will need to contact Veritas Tech Support for help. A migration utility is available from Flex Tech Support to generate a script for subdisk migration, if necessary.

Multiple LUNs for One Media MSDP Container

If you provision fewer than six media MSDP containers, you can allocate more than one LUN for some containers.

The storage shelves of the Flex 53xx Appliance can be populated with either 4-TB drives or 8-TB drives. The LUN size of 4-TB storage shelves is 40 TB and the LUN size for 8-TB storage shelves is 80 TB. If the volume size is more than 40 TB with 4-TB drive storage shelves, two LUNs will be concatenated to form the desired volume size.

For Flex Appliances, the default maximum volume size is 80 TB. You will need to create more than one VxVM volume to provision?

This default size will affect how many VxVM volumes/VxFS filesystems are created for a media container with a storage pool size greater than 80 TB. For example, to create a storage pool of 120 TB, two VxVM volumes will be created. How the two volumes are provisioned, however, will depend on the drive size of the storage shelf:

1. For a storage shelf populated with 4-TB drives, two LUNs will concatenated to form the first 80-TB VxVM volume and the second VxVM volume would be 40 TB sitting on a single LUN only. We introduced the Best Fit storage algorithm in Flex 2.0.1 that would create two file systems for 80 TB ask in a Flex 53xx 4-TB disk drive storage shelf as opposed to one file system as previously. The algorithm also takes care of choosing an appropriate LUN based on the storage size you specify.

2. For a storage shelf populated with 8-TB drives, the first 80-TB VxVM volume will be created on a single LUN and the remaining 40-TB VxVM volume will be provisioned out of another LUN.

The above two storage allocations are the default behavior, and the I/O performance should be comparable.

Tuning MSDP-Direct Cloud Tier

Beginning with version 8.3, NetBackup added a significant enhancement to the cloud tier offering, MSDP-C. The new enhancement not only simplifies cloud tier management, but also improves cloud tier performance. You no longer need to create a separate CloudCatalyst instance as the gateway for uploading data to cloud storage. This enhancement enables a single media server instance to configure multiple storage logical storage units (LSUs), including one local LSU and zero or more cloud LSUs. The LSU can be either regular or WORM storage. With NetBackup 8.3.0.1 and 9.0, WORM is only supported in local LSUs, and support for WORM storage in cloud LSUs with AWS Object Lock is available only in NetBackup 9.1.

Several tuning parameters have been introduced to let you customize the MSDP to meet the performance requirement of different workloads. These parameters are configured in the cloud.json file available at: <Storage>/etc/puredisk/cloud.json, where <Storage> should be something similar to the following path directory: "/var/lib/docker-veritas-plugin/vemLY_msdpdata-0_vxosdg"

One of the most important tuning parameters that can significantly impact uploading performance is UseMemForUpload. If the parameter is set to true, the upload cache directory is mounted in memory as tmpfs. The parameter is set to true by default when adding a cloud LSU. If there is enough memory available, then uploading to the cloud will go through memory; otherwise, cloud uploading will use disk cache, which is significantly slower. The rest of this section will cover how to tune the new parameters to ensure memory uploading occurs.

To ensure memory is used for cloud uploads, you must carefully tune the following four parameters together:

1. UsableMemoryLimit
2. MaxCacheSize
3. UploadCacheGB
4. MaxCloudCacheSize

UsableMemoryLimit—This parameter specifies the maximum amount of memory usable for the Fingerprint (FP) cache and cloud upload cache. The parameter is expressed as the percentage of physical RAM on the appliance. The default is 80 percent, which ensures at least 20 percent of the physical RAM is reserved for other processes and instances running on the system, such as NetBackup and kernel processes. The following relationship must be held to ensure memory is used for cloud uploads:

(MaxCacheSize + MaxCloudCacheSize + UploadCacheGB)/ (system RAM size) <= UsableMemoryLimit.

By default, the MaxCacheSize and MaxCloudCacheSize in a media server instance are set at 50 percent and 20 percent respectively, which leaves 10 percent of RAM for tpmfs. The default setting should work fine if there is only one local and one remote cloud storage configured on the Flex Appliance.

There maybe one or more MSDP instances on a Flex Appliance, and each instance may have one local and zero or more cloud LSUs. Each local LSU has its own MaxCacheSize and each remote LSU has its own MaxCloudCacheSize and UploadCacheGB (also called "Cloud in Memory upload cache size"). All the media server instances configured on the appliance need to share the 80 percent UsableMemoryLimit. To ensure memory is used for cloud upload, the above formula must hold for each individual instance. The sum of all MaxCacheSize, MaxCloudCacheSize and UploadCacheGB configured in all MSDP instances also should not exceed the 80 percent of the system RAM to ensure memory upload and avoid potential memory starvation.

MaxCacheSize—The recommended MaxCacheSize for a media server instance changed to 50 percent in Flex 2.0.1 (down from 60 percent). This change ensures the Flex Appliance performance with a single MSDP instance with or without MSDP-C is on par with a NetBackup Appliance. MaxCacheSize tuning as described in the section, "Media MSDP Instance Memory Tuning" is still applicable and must be followed with or without the remote LSU configuration.

UploadCacheGB—This parameter is set in the file <STORAGE>/etc/puredisk/cloud.json, and the default is 12 (that is, 12 GB). The default value for this parameter is set in the contentrouter.cfg parameter, CloudUploadCacheSize. The location of this upload cache can be in tmpfs or on local disk. If the parameter UseMemForUpload is false, the writes will go to local disk. If the parameter is true, a tmpfs will be created in memory and tmpfs will be used for cloud upload if there is enough memory available for tmpfs. For each cloud LSU, this parameter should be set to larger than the following:

(Max concurrent upload stream number) * MaxFileSizeMB *2

The MaxFileSizeMB is set in the cloud.json file and the default is 64 (that is, 64 MB).

MaxCloudCacheSize—This parameter is used to configure the amount of cache that can be used for temporarily caching the FPs needed for uploading to cloud storage. The purpose of the cache is similar to the FP cache configured for traditional client-side deduplication. Cache entries are not permanent and the content will be replaced by the next set of jobs once the previous jobs are completed. The size is specified as the percentage of the total physical RAM on the system and the default value is 20 percent. All cloud LSUs need to share this 20 percent of RAM.

Considerations when More Than One Media Instance Has at least One Cloud LSU Configured

As mentioned earlier, the following formula must be held for each individual media/MSDP instance:

"MaxCacheSize + MaxCloudCacheSize + (UploadCacheGB)/ (system RAM size) <= UsableMemoryLimit"

Most important, the 80 percent UsableMemoryLimit needs to be shared among all the media server instances configured on the appliance. Similarly, the 20 percent MaxCloudCacheSize needs to be shared by all cloud LSUs configured on the appliance.

If the workload and "max concurrent write stream number" for each cloud LSU is roughly the same, then the 20 percent of MaxCloudCacheSize can be divided equally among the cloud LSU instances configured on the system. Similarly, the 12 GB of UploadCacheSize can be divided equally as well. If the MaxCacheSize setting for each media server instance is also the same, then the 80 percent of UsableMemoryLimit can also be divided equally among the media/MSDP instances.

Let's illustrate the above with the following use case example:

1. A single node Flex 5340 Appliance has the following configuration:
   a. 768 GB of RAM
   b. 720 TB storage (three storage shelves populated with 4-TB drives)
2. There are three media server instances, each with one local and one cloud LSU
   a. The size of each local LSU is 240 TB.
   b. Each MSDP-C instance will handle at most 30 concurrent upload streams and the average size of the streams is 100 GB. In other words, the maximum concurrent upload images size is about 3 TB.

With the above specifications, Table 3 shows the tuning parameter settings for each of the four instances.

Table 3. Tuning Parameter Settings for Media Server Instances

Parameter/Instance	Instance 1	Instance 2	Instance 3
MaxCacheSize a	16%	16%	16%
MaxCloudCacheSize b	7%	7%	7%
UploadCacheGB c	4	4	4
UsableMemoryLimit d	26%	26%	26%

a. The MaxCacheSize is set to 16 percent, which is about 120 GB of the 768 GB of system RAM. Each local LSU storage size is 240 TB, which needs 240 GB of RAM to run the instance, according to the rule of thumb. Out of the 240 GB of RAM, we recommend configuring half of it, or 120 GB, for MSDP FP caching. The 120 GB is roughly 16 percent of the total RAM on the Flex Appliance, which is why the MaxCacheSize is set to 16 percent in Table 3.

b. The MaxCloudCacheSize default is 20 percent, which needs to be shared among the three MSDP-C instances. Because we assume the maximum upload image sizes are similar among the three cloud instances, we can divide the 20 percent equally between the three cloud instances, resulting in about 7 percent each.

If the upload image sizes among the cloud instances are different, you can fine-tune the MaxCloudCacheSize setting following these steps:

1. Maximum concurrent upload image size in KB = (Average stream size in KB * maximum concurrent upload streams)
2. # of total Fingerprints = (result from step 1 above)/128KB . . . . . .
   Where 128 KB is the default data segment size (DefaultSegmentSize) is defined in contentrouter.cfg.
3. The size of MaxCloudCacheSize in Bytes = (result from step 2 above * 48(Bytes)) . . . . . .
   Where 48 Bytes is the size of RAM required to cache a FP in memory,
4. The MaxCloudCacheSize = (convert the result from step 3 above to GB) / Total RAM in GB * 100

c. The UploadCacheGB default is 12 GB, divided equally among the three MSDP-C instances to get the 4 GB per instance amount.

If the # of concurrent upload streams is different between the cloud LSU instances, you can use the following formula to calculate he UploadCacheGB for each instance:

(Max concurrent upload stream number) * MaxFileSizeMB *2

So, the 4 GB of UploadCacheGB can support up to 32 concurrent upload streams.

d. The UsableMemoryLimit default is 80 percent, divided equally among the three MSDP-C instances to get the 26 percent figure per instance.

If the aggregate MaxCacheSize is lower than 50percent of the RAM size, then the aggregate MaxCloudCacheSize can be larger than 20 percent and the aggregate UploadCacheGB can be greater than 12 GB as long as the 80 percent of aggregate UsableMemoryLimit is maintained.

For example, assume the RAM size of a Flex 5340 Appliance is 1.5 TB and the total storage capacity is 960 TB. Then according to the best practice guide of allocating MaxCacheSize, we would be allocating, in aggregate, approximately 480 GB or 32 percent of the RAM for the MaxCacheSize. In other words, MaxCloudCacheSize and UploadCacheGB get an additional 18 percent of RAM allocated for improving performance if necessary without violating the 80 percent UsableMemoryLimit.

For more details, refer to the Veritas NetBackup Deduplication Guide.

Appendix

Flex Tuning Parameters

This section lists all the kernel, Veritas File System and NetBackup tunings that were implemented in the Flex 1.2 and 1.3 releases. A couple of tunings that were discovered too late to be included in Flex 1.2 are listed in the manual tuning sections below.

Default Tunings in Flex 1.2

Parameter	Default Value	OS/VxFS/NetBackup
vm.max_map_count	262140	OS
Transparent Hugepage	Disabled	OS
vm.min_free_kbytes	2097152	OS
vm.swappiness	10	OS
vm.overcommit_memory	2	OS
vm.overcommit_ratio	90	OS
Ring buffer Size Rx and TX	1 Gb NIC = 1024 10 Gb NIC = 2048	OS
vx_write_flush_behind_enabled	1	VxFS
max_seqio_extent_size	16384	VxFS
Semaphores on Primary instance	400 307200 32 1024	Primary Instance
MaxConnections	8192	MSDP Instance

Default Tunings in Flex 1.3

In addition to the tuning parameters in Flex 1.2, the following tunings are included in Flex 1.3:

Parameter	Default Value	OS/VxFS/NetBackup
kernel.numa_balancing	0	OS
vm.overcommit_ratio	100	OS

Default Tunings in Flex 2.0 and Flex 2.0.1

Beginning in Flex 2.0.x, the following VxFS tuning parameters were added to improve I/O performance:

Parameter	Default Value	OS/VxFS/NetBackup
read_nstream	8	VxFS
write_pref_io	209152	VxFS
max_diskq	8388608	VxFS
dalloc_enable	0	VxFS

The above four tunings should improve I/O subsystem read/write performance. For more information about the effect of the above four parameters, check the man page of the command vxtunefs.

To change the parameters dynamically:
vxtunefs -o read_nstream=8 <mount_point>
vxtunefs -o write_pref_io=2097152 <mount_point>
vxtunefs -o max_diskq=8388608 <mount_point>
vxtunefs -o dalloc_enable=0 <mount_point>

To make the changes persistent, modify /etc/vx/tunefstab and add the following line
>>>
system_default read_nstream=8
system_default write_pref_io=2097152
system_default max_diskq=8388608
system_default dalloc_enable=0

Manual Tunings Needed in Flex 1.2

This section lists all the kernel, Veritas File System and NetBackup tunings that were implemented in the Flex 1.2 and 1.3 releases. A couple of tunings that were discovered too late

1. numa_balancing

Turn off numa_balancing on the Flex Appliance to avoid unnecessary cost in memory migration and memory swapping.

You can do so in the Flex host as shown below:

# vi /etc/sysctl.conf
kernel.numa_balancing = 0
# sysctl -p /etc/sysctl.conf

2. overcommit_ratio

Increase the overcommit_ratio from default 90 to 100 to handle higher bursts in virtual memory usage.

This can be changed at You can do so in the Flex host as shown below:

```
# vi /etc/sysctl.conf
vm.overcommit_ratio = 100
# sysctl -p /etc/sysctl.conf
```

Manual Tunings Needed in Flex 1.2 and Flex 1.3

1. MaxCacheSize

Change the MaxCacheSize as shown below:

```
ssh appadmin@<msdp_instance_name>
$ sudo /usr/openv/pdde/pdag/bin/pdcfg --write /mnt/msdp/vol0/etc/puredisk/contentrouter.cfg --section CACHE --
option MaxCacheSize --value 60%
```

Change the MaxCacheSize as shown below:

```
$ sudo /usr/openv/pdde/pdag/bin/pdcfg --read /mnt/msdp/vol0/etc/puredisk/contentrouter.cfg --section CACHE --option MaxCacheSize
```

Restart the pdde-storage process with the following commands to make the change take effect:

```
$ sudo /etc/init.d/pdde-storage force-stop
$ sudo /etc/init.d/pdde-storage start
```

2. Swappiness

The default value is 10. You can manually change the value to 1 if you see high swap usage and page swap in and out. Monitor the performance after changing the value to 1 because this change can cause performance degradation in some cases.

You can do so in the Flex host as shown below:

```
# vi /etc/sysctl.conf
vm.swappiness= 1
# sysctl -p /etc/sysctl.conf
```

Manual Tunings Needed in Flex 2.0

1. Apply hotfix

Flex VxFS patch and MSDP EEB bundle for Flex2.0 running NetBackup 8.2 or NetBackup 8.3.0.1 container. The hotfix is needed to improve mixed (read/write) workload performance. For more details on applying the hotfix, see the technote:

2. Net.core.somaxconn

Starting from NetBackup 9.0.1, the net.core.somaxconn changed from 128 to 1024 by default.

This kernel parameter determines the maximum number of backlogged connections allowed for each TCP port. The default value is 128, but we recommend changing it to 1024 to avoid a "connection refused" error between clients and the appliance.

Prior to NetBackup 9.0.1, you can apply the tuning manually, as shown below.

To tune the value for the NetBackup instance on the Flex Appliance host:

```
# cd /mnt/data/infra/profiles/instances
# vi izTAR_8-2.json
[...]
"sysctls": [
"kernel.sem=400 307200 32 1024",
"net.ipv4.tcp_keepalive_intvl=10",
"net.ipv4.tcp_keepalive_probes=5",
"net.ipv4.tcp_keepalive_time=900",
"net.core.somaxconn=1024"
],
[...]
```

Restart the NetBackup instance and verify the tuning change in the NetBackup instance:

```
# cat /proc/sys/net/core/somaxconn
1024
```

3. AllocationUnitSize

The default for this MSDP parameter changed to 8 MiB with a Media MSDP container running NetBackup 8.3.0.1; the default was 2 MiB. If you are running a NetBackup container prior to version 8.3.0.1, you need to set this parameter to 8 MiB manually, especially if the appliance has 1.5 TB of RAM installed.

Change the AllocationUnitSize in the MSDP container:

```
# ssh appadmin@<msdp_instance_name>
$ sudo /usr/openv/pdde/pdag/bin/pdcfg --write  /mnt/msdp/vol0/etc/puredisk/contentrouter.cfg --section CACHE --
option AllocationUnitSize --value 8MiB
```

Check the changed AllocationUnitSize as shown below:

```
$ sudo /usr/openv/pdde/pdag/bin/pdcfg --read /mnt/msdp/vol0/etc/puredisk/contentrouter.cfg --section CACHE --
option AllocationUnitSize
```

Restart the pdde-storage process with the following commands:

```
$ sudo /etc/init.d/pdde-storage force-stop
$ sudo /etc/init.d/pdde-storage start
```

4. vm.extfrag_threshold

This kernel parameter impacts whether the kernel does memory compaction or memory reclaim to satisfy a high-order allocation; the default is 500. The default should work fine for most installations. If the appliance performance is suffering frequently due to memory fragmentation, however, try increasing this value to 1,000 to free memory through reclaim.

You can do so in the Flex host as shown below:

```
# vi /etc/sysctl.conf
vm.extfrag_threshold = 1000
# sysctl -p /etc/sysctl.conf
```

5. read_nstream

This vxfs parameter impacts the file system read ahead performance. Increasing this value can improve rehydration performance. The default value was 1 for Flex releases before 2.0.1. The new recommendation is to set this parameter to 8 to improve read performance.

To change it dynamically:

```
# vxtunefs -o read_nstream=8 <mountpoint_of_msdp_data_volume>
```

Versions

Parameter	Date	Author	Key updates
1.0	Nov 2021	Su-jin Chan, Angela Ellingsen, Rachel Zhu	Original document.

About Veritas

Veritas Technologies is a global leader in data protection and availability. Over 80,000 customers—including 87 percent of the Fortune Global 500—rely on us to abstract IT complexity and simplify data management. The Veritas Enterprise Data Services Platform automates the protection and orchestrates the recovery of data everywhere it lives, ensures 24/7 availability of business-critical applications, and provides enterprises with the insights they need to comply with evolving data regulations. With a reputation for reliability at scale and a deployment model to fit any need, Veritas Enterprise Data Services Platform supports more than 800 different data sources, over 100 different operating systems, more than 1,400 storage targets, and more than 60 different cloud platforms. Learn more at www.veritas.com. Follow us on Twitter at @veritastechllc.

VERITAS

2625 Augustine Drive
Santa Clara, CA 95054
+1 (866) 837 4827
veritas.com

For global contact information visit:
veritas.com/company/contact

Copyright © 2021 Veritas Technologies LLC. All rights reserved. Veritas, the Veritas Logo, and NetBackup are trademarks or registered trademarks of Veritas Technologies LLC or its affiliates in the U.S. and other countries. Other names may be trademarks of their respective owners.

V1452 11/21