Veritas InfoScale™ 8.0 SmartIO for Solid-State Drives Solutions Guide - Linux

VERITAS™

Last updated: 2021-12-21

Legal Notice

Copyright © 2021 Veritas Technologies LLC. All rights reserved.

Veritas and the Veritas Logo 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.

This product may contain third-party software for which Veritas is required to provide attribution to the third-party (“Third-Party Programs”). Some of the Third-Party Programs are available under open source or free software licenses. The License Agreement accompanying the Software does not alter any rights or obligations you may have under those open source or free software licenses. Refer to the third-party legal notices document accompanying this Veritas product or available at:
https://www.veritas.com/about/legal/license-agreements

The product described in this document is distributed under licenses restricting its use, copying, distribution, and decompilation/reverse engineering. No part of this document may be reproduced in any form by any means without prior written authorization of Veritas Technologies LLC and its licensors, if any.

THE DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID. VERITAS TECHNOLOGIES LLC SHALL NOT BE LIABLE FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS DOCUMENTATION. THE INFORMATION CONTAINED IN THIS DOCUMENTATION IS SUBJECT TO CHANGE WITHOUT NOTICE.

The Licensed Software and Documentation are deemed to be commercial computer software as defined in FAR 12.212 and subject to restricted rights as defined in FAR Section 52.227-19 "Commercial Computer Software - Restricted Rights" and DFARS 227.7202, et seq. "Commercial Computer Software and Commercial Computer Software Documentation," as applicable, and any successor regulations, whether delivered by Veritas as on premises or hosted services. Any use, modification, reproduction release, performance, display or disclosure of the Licensed Software and Documentation by the U.S. Government shall be solely in accordance with the terms of this Agreement.

Veritas Technologies LLC
2625 Augustine Drive
Santa Clara, CA 95054
http://www.veritas.com

Technical Support

Technical Support maintains support centers globally. All support services will be delivered in accordance with your support agreement and the then-current enterprise technical support policies. For information about our support offerings and how to contact Technical Support, visit our website:
https://www.veritas.com/support

You can manage your Veritas account information at the following URL:
https://my.veritas.com

If you have questions regarding an existing support agreement, please email the support agreement administration team for your region as follows:

Worldwide (except Japan)	CustomerCare@veritas.com
Japan	CustomerCare_Japan@veritas.com

Documentation

Make sure that you have the current version of the documentation. Each document displays the date of the last update on page 2. The latest documentation is available on the Veritas website:
https://sort.veritas.com/documents

Documentation feedback

Your feedback is important to us. Suggest improvements or report errors or omissions to the documentation. Include the document title, document version, chapter title, and section title of the text on which you are reporting. Send feedback to:
infoscaledocs@veritas.com

You can also see documentation information or ask a question on the Veritas community site:
http://www.veritas.com/community/

Veritas Services and Operations Readiness Tools (SORT)

Veritas Services and Operations Readiness Tools (SORT) is a website that provides information and tools to automate and simplify certain time-consuming administrative tasks. Depending on the product, SORT helps you prepare for installations and upgrades, identify risks in your datacenters, and improve operational efficiency. To see what services and tools SORT provides for your product, see the data sheet:
https://sort.veritas.com/data/support/SORT_Data_Sheet.pdf

Contents

Chapter 1 Introducing SFHA Solutions SmartIO 7

About SmartIO for solid-state drives 7
About SmartIO in an SFHA environment 8
About SmartIO in an Active/Active cluster environment 9
About SmartIO in the Linux virtualized environment 9
About the SmartIO caching profiler tool 11

Chapter 2 Using the SmartIO feature: use cases 12

About SmartIO read caching for applications running on VxVM volumes 12
Required configuration for SmartIO read caching for VxVM volumes 14
Automatic caching for VxVM volumes 14
Setting up SmartIO read caching for VxVM volumes 15
Verifying the VxVM cache area and monitoring the caching 16
About SmartIO read caching for applications running on VxFS file systems 19
Required configuration for SmartIO read caching for a VxFS file system 20
Automatic caching for VxFS file systems 20
Setting up SmartIO read caching for VxFS file systems 21
Verifying the VxFS cache area and monitoring the caching 22
Customizing the caching behavior 26
About SmartIO caching on SSD devices exported by FSS 27
Status of cache areas when nodes leave or join the cluster 28
Setting up cache areas using SSDs exported by FSS 29
About SmartIO write-back caching for applications running on VxFS file systems- DLV 11 to 13 30
Required configuration for SmartIO write-back caching for a VxFS file system 31
Setting up SmartIO write-back caching for VxFS file systems 31
Verifying the VxFS cache area and monitoring the caching (writeback mode) 33
About SmartIO FEL-based writeback caching for applications running on VxFS file systems- DLV 14 and later 36

Required configuration for FEL-based write-back caching for a VxFS file system 38
Setting up FEL-based write-back caching for VxFS file systems 38
Verifying the VxFS cache area and monitoring FEL-based write-back caching 40
About multiple SmartIO cache areas for read and write-back caching on VxFS file systems 42
About the `smartiocache` option 46
Converting VxFS cache areas from one type to another 46
Setting up multiple cache areas on a system 47
Verifying the VxFS cache areas 49
About SmartIO caching for Oracle databases on VxFS file systems 52
Prerequisites and configuration for using the SmartIO plug-in for Oracle 52
Setting default SmartIO caching policies for a database running on a VxFS file system 53
Setting SmartIO caching policies for database objects 55
Pinning and unpinning database objects 56
Enabling and disabling caching for the database 56
Listing cache policy details for the database 57
Listing cache statistics for the database 58
About SmartIO caching for databases on VxVM volumes 59
Applying a SmartIO database caching template for a VxVM volume 59
Technology Preview: Distributed SmartIO in Veritas InfoScale storage environments 63

Chapter 3 Administering SmartIO 69

Creating a cache area 69
Displaying information about a cache area 73
Enabling or disabling caching for a data object 76
Enabling or disabling caching for a file system 76
Enabling or disabling caching for a data volume 77
Adding a device to the cache area 78
Pausing caching from a volume to a cache area 78
Removing a device from the cache area 78
Destroying a cache area 79
Setting the attributes of the VxVM cache area 80
Setting or changing the caching mode for a VxFS cache area 81
Flushing dirty data from a writeback cache area 82

Tuning the writeback caching 82
Viewing the SmartIO cache statistics 84
Viewing the detailed caching stats for a VxVM cache area 86
Viewing the detailed caching stats for a VxFS cache area 87

Chapter 4 Troubleshooting and error handling 90

Support for a persistent or 'warm' VxVM cache 90
Primary volume failure with a stale cache could cause possible data corruption 91
Migrating a cache during HA failover is not supported 91
Cache area is lost after a disk failure (3158482) 91
Cache is not online after a reboot 92
Recovering the write-back cache after a node failure 92

Appendix A Command reference 94

SmartIO command reference 94

Chapter 1

Introducing SFHA Solutions SmartIO

This chapter includes the following topics:

- About SmartIO for solid-state drives
- About SmartIO in an SFHA environment
- About SmartIO in an Active/Active cluster environment
- About SmartIO in the Linux virtualized environment
- About the SmartIO caching profiler tool

About SmartIO for solid-state drives

Solid-state drives (SSDs) are devices that do not have spinning disks. Today's solid-state technologies, such as DRAM and NAND flash, provide faster data access, are more efficient, and have a smaller footprint than traditional spinning disks. The data center uses solid-state technologies in many form factors: in-server, all flash arrays, all flash appliances, and mixed with traditional HDD arrays. Each form factor offers a different value proposition. SSDs also have many connectivity types: PCIe, FC, SATA, and SAS.

Due to the current cost per gigabyte of SSD devices, the best value of SSDs is not as high capacity storage devices. The benefit of adopting SSDs is to improve performance and reduce the cost per I/O per second (IOPS). Data efficiency and placement is critical to maximizing the returns on any data center's investment in solid state.

The SmartIO feature of Storage Foundation and High Availability Solutions (SFHA Solutions) enables data efficiency on your SSDs through I/O caching. Using SmartIO to improve efficiency, you can optimize the cost per IOPS. SmartIO does not require

in-depth knowledge of the hardware technologies underneath. SmartIO uses advanced, customizable heuristics to determine what data to cache and how that data gets removed from the cache. The heuristics take advantage of SFHA Solutions' knowledge of the characteristics of the workload.

SmartIO uses a cache area on the target device or devices. The cache area is the storage space that SmartIO uses to store the cached data and the metadata about the cached data. The type of the cache area determines whether it supports VxFS caching or VxVM caching. To start using SmartIO, you can create a cache area with a single command, while the application is online.

When the application issues an I/O request, SmartIO checks to see if the I/O can be serviced from the cache. As applications access data from the underlying volumes or file systems, certain data is moved to the cache based on the internal heuristics. Subsequent I/Os are processed from the cache.

SmartIO supports read and write caching for the VxFS file systems that are mounted on VxVM volumes, in several caching modes and configurations. SmartIO also supports block-level read caching for applications running on VxVM volumes.

See "About SmartIO read caching for applications running on VxVM volumes" on page 12.

See "About SmartIO read caching for applications running on VxFS file systems" on page 19.

See "About SmartIO write-back caching for applications running on VxFS file systems- DLV 11 to 13 " on page 30.

See "About SmartIO FEL-based writeback caching for applications running on VxFS file systems- DLV 14 and later" on page 36.

See "About SmartIO caching for Oracle databases on VxFS file systems " on page 52.

See "About SmartIO caching for databases on VxVM volumes" on page 59.

About SmartIO in an SFHA environment

In a clustered environment, the SmartIO cache is local to each node in the cluster. The cache area cannot be brought offline from one node and then brought online on a different node.

When SmartIO read caching is used in an Active/Passive environment such as with SFHA, you can deport and import the data volumes and file systems to another node. The SmartIO cache is not moved to the other node. Caching starts on the new node with the local SmartIO cache area there. SmartIO does not provide the capability to migrate the cache in a high availability cluster during a failover.

For VxFS write-back caching, failing over the disk group is not recommended. In the case of local mount, if there is no cache mirror, the disk copy of the file can be incomplete or stale if the node fails. Veritas recommends using SFCFSHA instead, which provides data reflection of the cache.

About SmartIO in an Active/Active cluster environment

When SmartIO read caching is used in Active/Active environment such as SF Oracle RAC, caching starts on all nodes with the local SmartIO cache area on each node. The cache areas can be brought online or offline exclusively on each node.

SmartIO VxFS write-back caching is not currently supported for SF Oracle RAC.

About SmartIO in the Linux virtualized environment

In the Linux virtualized environment, when you install Veritas InfoScale Solutions in the guest, you can use SmartIO to cache data onto an SSD or any other supported fast device.

SmartIO caching does not support live migration of the guest in KVM and RHEV environments.

For VMware, SmartIO does support vMotion if DMP for VMware (SmartPool) is enabled in the ESXi hypervisor.

Storage Foundation for Oracle RAC is not supported in the Linux virtualized environment.

The following tables show how SmartIO can be used in the Linux virtualized environments.

Table 1-1 shows how SmartIO can be used in the KVM environment.

Table 1-1 Linux: SmartIO support in KVM

Configuration in guest:	Configuration in host:	Caching takes place:	VxVM read caching	VxFS read caching	VxFS writeback caching
SF	any (SF or SFCFSHA)	in the guest	Yes	Yes	Yes
SFHA	any (SF or SFCFSHA)	in the guest	Yes	Yes	Yes

Table 1-1 Linux: SmartIO support in KVM *(continued)*

Configuration in guest:	Configuration in host:	Caching takes place:	VxVM read caching	VxFS read caching	VxFS writeback caching
SFCFSHA	any (SF or SFCFSHA)	in the guest	Yes	Yes	Yes
Any	SF	in the host	Yes	Yes	Yes
Any	SFCFSHA	in the host	Yes	Yes	Yes

Table 1-2 shows how SmartIO can be used in the RHEV environment.

Table 1-2 Linux: SmartIO support in RHEV

Configuration in guest:	Configuration in host:	Caching takes place:	VxVM read caching	VxFS read caching	VxFS writeback caching
SF	any (SF or SFCFSHA)	in the guest	Yes	Yes	Yes
SFHA	any (SF or SFCFSHA)	in the guest	Yes	Yes	Yes
SFCFSHA	any (SF or SFCFSHA)	in the guest	Yes	Yes	Yes
Any	SF	in the host	Yes	Yes	Yes
Any	SFCFSHA	in the host	Yes	Yes	Yes

Table 1-3 shows how SmartIO can be used in the VMware environment.

Table 1-3 Linux: SmartIO support in VMware

Configuration in guest:	Configuration in host:	Caching takes place:	VxVM read caching	VxFS read caching	VxFS writeback caching
SF	DMP for VMware (Optional)	in the guest	Yes	Yes	No
SFHA	DMP for VMware (Optional)	in the guest	Yes	Yes	No
SFCFSHA	DMP for VMware (Optional)	in the guest	Yes	Yes	No

For more information about configuring Veritas InfoScale Solutions in the Linux Virtualization environment, see the *Veritas InfoScale™ Solutions Virtualization Guide for Linux*.

About the SmartIO caching profiler tool

The Smartassist tool analyzes the I/O on the specified target in the system for the specified time, and calculates the optimal cache size for the workload. To run the tool from Veritas InfoScale Operations Manager (VIOM), refer to VIOM administrator's guide.

Smartassist tool supports following targets:

- List of device paths
- List of VxVM or LVM volumes paths
- List of VxVM or LVM volumes paths
- List of filesystem mount points
- Oracle/Sybase/DB2 database instance
- List of shared volumes/diskgroups of VxVM
- List of CFS mount points

The tool works in two phases:

Start phase	The tool traces the I/O on the specified target for the specified time and stores the trace output in the specified directory. In case of disk devices, which have multiple paths, it is advisable to specify all the paths to get correct results. Using VxVM or LVM volume devices, in such cases, is more convenient. By default, the start phase runs for 3600 seconds.
Analyze phase	The tool parses the output generated in the start phase and uses the SmartIO algorithms to calculate the optimal cache size. The optimal cache size, read hits, latency gain, performance forecast and share in cache for each target are displayed on the terminal.

For information on downloading and installing the tool:

https://sort.veritas.com/dc_download/readme/smartassist

Chapter 2

Using the SmartIO feature: use cases

This chapter includes the following topics:

- About SmartIO read caching for applications running on VxVM volumes
- About SmartIO read caching for applications running on VxFS file systems
- About SmartIO caching on SSD devices exported by FSS
- About SmartIO write-back caching for applications running on VxFS file systems- DLV 11 to 13
- About SmartIO FEL-based writeback caching for applications running on VxFS file systems- DLV 14 and later
- About multiple SmartIO cache areas for read and write-back caching on VxFS file systems
- About SmartIO caching for Oracle databases on VxFS file systems
- About SmartIO caching for databases on VxVM volumes
- Technology Preview: Distributed SmartIO in Veritas InfoScale storage environments

About SmartIO read caching for applications running on VxVM volumes

SmartIO supports block-level read caching for Veritas Volume Manager (VxVM) volumes. This type of SmartIO caching primarily supports the applications that run directly over raw volumes. For example, database instances running directly over

raw volumes. Volume-level caching can also be used in cases where VxFS caching cannot be used. SmartIO only supports read caching at the volume level.

The SmartIO cache typically resides on one or more SSD devices or other fast devices. SmartIO accelerates the read I/O performance because the application read I/Os are serviced from the SSD-based cache rather than the standard storage.

SmartIO does not require complex configuration to set up caching. You simply set up a cache area, which is the storage space for the cached data and metadata about the cache. For volume-level read caching, the cache area has the VxVM type. A single VxVM cache area is used per system. By default, the SmartIO cache area enables automatic caching for all VxVM volumes on the system. If you prefer, you can configure the cache area as noauto. For a noauto cache area, you must explicitly enable SmartIO read caching for the VxVM volumes. The configuration of the cache area is persistent.

See “Automatic caching for VxVM volumes” on page 14.

For each VxVM volume on which caching is enabled, SmartIO determines which data to cache or to evict from the cache. SmartIO uses its knowledge of the workload to optimize its use of the cache.

The SmartIO feature supports only one VxVM cache area on a system. For each system, all VxVM volumes that are cached share a single cache area of VxVM type. Multiple VxVM cache areas are not supported, although the same system can have both a VxFS cache area and a VxVM cache area.

A cache area is private to each node in a cluster. The cache contents are not shared across the nodes in the cluster.

A SmartIO cache preserves cache coherency at the volume level. If the cache device becomes inaccessible while caching is enabled, the application continues to function normally. However, application performance may be reduced.

In a Cluster Volume Manager (CVM) environment, SmartIO uses a cache coherency protocol to keep cache areas on multiple nodes coherent when writes are made to a shared volume. A write on the data volume invalidates the contents on the cache area of other nodes. The cache coherency protocol uses the Group Lock Manager (GLM) module for communication. When the cache is initially getting populated, the cache coherency protocol creates a small performance overhead in the write I/O path.

The data in the read cache is not persistent by default. In the case of a planned system reboot, you can choose to create a warm cache.

See “Support for a persistent or 'warm' VxVM cache” on page 90.

Required configuration for SmartIO read caching for VxVM volumes

You can set up SmartIO for read caching for VxVM volumes with the following configurations:

- A Storage Foundation RAC (SFRAC) cluster or a Storage Foundation Cluster File System High Availability (SFCFSHA) cluster. A cache area cannot be on a shared volume. The VxVM cache area must be configured as local to each node, as shared access of cache area is not supported.
- A Storage Foundation High Availability (SFHA) cluster. The VxVM cache area must be configured as local to each node, as shared access of cache area is not supported.
  See "About SmartIO in an SFHA environment" on page 8.
- A stand-alone Storage Foundation system.

The volumes to be cached must have the disk group version 190 or above.

The devices used for the cache area have the following characteristics:

- Utilize faster devices such as solid-state drives (SSDs) supported by Veritas Volume Manager (VxVM) to accelerate read IO performance. However, any devices supported by VxVM can be used for the cache area.
- Must be initialized for use with VxVM, and must have the `cdsdisk` format.

Automatic caching for VxVM volumes

The association type of a cache area indicates whether or not automatic caching is enabled for the system. The association type attribute for the VxVM cache area is persistent. The association type can be one of the following:

- `auto` attribute (default)
  The cache area is enabled for automatic caching. All VxVM data volumes on the system are cached unless you explicitly disable caching for that volume. You do not need to explicitly enable caching on a volume.
  SmartIO does not support caching RAID-5 volumes and DCO volumes. Also, SmartIO does not enable automatic caching for volumes used for logging and cache objects including Storage Replication Logs (SRLs), Data Change Maps (DCMs), and volumes used for space-optimized snapshot cache objects.
  By default, a VxVM cache area has the `auto` attribute.
- `noauto` attribute
  The cache area is not enabled for automatic caching. No volumes are cached automatically. You must explicitly enable caching for each volume that you want cached. You do not need to explicitly disable a volume for caching, except to exclude a volume that was previously enabled. You can enable caching when

you create a volume. You can also selectively enable or disable read caching on an existing VxVM volume, without quisceing the I/O.

Setting up SmartIO read caching for VxVM volumes

In read mode, the SmartIO feature caches the VxVM I/Os. To set up SmartIO for read caching for a VxVM volume, simply create the cache area.

Setting up SmartIO read caching for VxVM volumes

1. Create a VxVM type cache area on an SSD device, using one of the following commands:

   - Specify one or more devices using the disk access name (*daname*) of the device. The device should be initialized for VxVM use and have the `cdsdisk` format.

     ```
     # sfcache create -t VxVM [size] daname[...] \
     [cacheline_size=cacheline_size] [--auto|--noauto]
     [--nostripe|ncols=N] [cachearea_name]
     ```

     Where:
     daname specifies the disk access name of the device on which to create the cache area.
     cachearea_name specifies a customized name for the cache area. If unspecified, the SmartIO feature automatically generates a name for the cache area.
     size specifies the size for the cache area. By default, the cache area uses all of the available space on the device.
     `--noauto|--auto` specifies the type of cache area. The default is `--auto`.
     `--nostripe|ncols=n` specifies the layout options for the cache area. By default, the cache area is created over a striped volume if two or more disks are specified. Use the `ncols=n` option to specify the number of columns for the striped volume. Use the `--nostripe` option to create the cache area on a concatenated volume over the specified disks.
     cacheline_size specifies the unit that SmartIO uses for caching. When the application I/O accesses the data, the SmartIO moves the data to the cache according to the cacheline size. Generally, you do not need to change the *cacheline_size*.
     For example:

     ```
     # sfcache create -t VxVM ssd0_0
     ```

- Alternatively, you can specify the name of a disk group and a volume to use for the cache area. In this case, you cannot specify a size: the cache area uses the entire volume.

  ```
  # sfcache create -t VxVM  [cacheline_size=cacheline_size] \
   [--noauto|--auto] dg/vol
  ```

  Where:
  dg/*vol* specifies the disk group name and volume name that you want to use for the cache area. SmartIO uses this name for the cache area.
  `--noauto|--auto` specifies the association type of the cache area. The default is `--auto`.
  For example:

  ```
  # sfcache create -t VxVM --auto ssd_dg/ssd_vol
  ```

2 Enable caching for the required VxVM volumes. If the cache area is auto, this step is not required. Caching is enabled by default when the SSD device comes online, for any VxVM volumes.

If the cache area is `noauto`, you must enable caching for the volumes to be cached.

```
# sfcache enable [--read] dg/vol
```

Where:

dg/*vol* specifies the disk group name and volume name of the volume that you want to cache.

For example:

```
# sfcache enable mydg/vol1
```

Verifying the VxVM cache area and monitoring the caching

After the SmartIO feature is configured, you can verify that the cache area is present and that caching is occurring.

To verify and monitor the cache area

1 Use the following command to display information about the cache areas on the system.

```
# sfcache list -l

Cachearea: sfcachearea_1
Assoc Type: AUTO
Type: VxVM
Size: 30.00g
Cacheline Size: 64.00k
Memory Size: 16.00m
State: ONLINE
Layout: CONCAT
Number of Columns: 0

ASSOCIATED DATA OBJECTS:

Volume: testdg/testvol1
Size: 500.00g
State: ENABLED
Kstate: ENABLED
Caching Mode: read

Volume: testdg/testvol2
Size: 500.00g
State: ENABLED
Kstate: ENABLED
Caching Mode: read
```

See “Displaying information about a cache area” on page 73.

2 Use the following command to display information about a specific cache area.

```
# sfcache list sfcachearea_1
Cachearea: sfcachearea_1
Assoc Type: AUTO
Type: VxVM
Size: 30.00g
Cacheline Size: 64.00k
Memory Size: 16.00m
State: ONLINE
Layout: CONCAT
Number of Columns: 0

ASSOCIATED DATA OBJECTS:

ASSOC DATAOBJECT NAME   CACHING-MODE  STATE    KSTATE
testdg/testvol1         read          ENABLED  ENABLED
testdg/testvol2         read          ENABLED  ENABLED
```

See “Viewing the SmartIO cache statistics ” on page 84.

3 To see statistics on the cache usage, use the following command:

```
# sfcache stat sfcachearea_1
```

```
                          HIT RATIO    ART(Hit)ms   ART(Miss)ms   BYTES
NAME                   %CACHE RD    WR     RD    WR     RD    WR     RD     WR
TYPE: VxVM
sfcachearea_1           13.43  91.24 94.20  0.142 0.819  0.414  0.798  15.31g 4.21g

ASSOCIATED DATA OBJECTS:
testdg/testvol1          6.10   90.00 96.00  0.141 0.459  0.348  0.448  6.77g  1.89g
testdg/testvol2          7.32   91.00 92.00  0.143 1.179  0.480  1.149  8.54g  2.31g
```

4 Use the following command to display information about the usage of exported SSDs by other nodes in the cluster. Run the command on the node for which you want to see the existing cache areas.

Note: Private cache areas of other nodes are not listed.

```
# sfcache list --all
Hostname : sys1
NAME              TYPE   SIZE       ASSOC-TYPE  STATE   FSTYPE   DEVICE
sfcachearea_1     VxVM   10.00g     AUTO        ONLINE    -      ibm_f90-0_0
sfcachearea_3     VxFS   10.00g     AUTO        ONLINE  reserve  ibm_f90-0_0

Hostname : sys2
NAME              TYPE   SIZE       ASSOC-TYPE  STATE   FSTYPE   DEVICE
sfcachearea_4     VxFS   20.00g     AUTO        ONLINE  reserve  ibm_f90-0_0
sfcachearea_5     VxVM   25.00g     AUTO        ONLINE    -      ibm_f90-0_0

Hostname : sys3
NAME              TYPE   SIZE       ASSOC-TYPE  STATE   FSTYPE   DEVICE
sfcachearea_2     VxFS   10.00g     AUTO        ONLINE  reserve  ibm_f90-0_0
```

About SmartIO read caching for applications running on VxFS file systems

Storage Foundation High Availability Solutions supports read caching on solid-state drives (SSDs) for applications running on VxFS file systems. In this scenario, application reads are satisfied from the cache whenever possible. As the application accesses the file system, the file system loads data from the disk into the cache.

Application writes go to the disk in the usual way. With each write, the file system synchronizes the cache to ensure that applications never see stale data. If a cache device fails, a file that is cached in read mode is completely present on the disk. Therefore, the cache failure does not affect the application I/Os for the file and the application I/Os continue without interruption.

Required configuration for SmartIO read caching for a VxFS file system

You can set up SmartIO for read caching for a VxFS file system with the following configurations:

- A Storage Foundation RAC (SF Oracle RAC) cluster, a Storage Foundation Cluster File System High Availability (SFCFSHA) cluster, or a Storage Foundation High Availability (SFHA) cluster, with private SSDs in each node.
  See "About SmartIO in an SFHA environment" on page 8.
- A standalone Storage Foundation system.
- Must be mounted on a VxVM volume.

Automatic caching for VxFS file systems

The SmartIO feature supports multiple VxFS cache areas on a system. The association type of a cache area indicates whether or not automatic caching is enabled for the system. The association type attribute for the VxFS cache area is persistent. The association type can be one of the following:

- `auto` attribute (default)

  The cache area is enabled for automatic caching. All file systems on the system are cached unless you explicitly disable caching for that file system. You do not need to explicitly enable caching on a file system. By default, a VxFS cache area has the `auto` attribute.

- `noauto` attribute

  The cache area is not enabled for automatic caching. No file systems are cached automatically. You must explicitly enable caching for each file system that you want cached. You do not need to explicitly disable a file system for caching, except to exclude a file system that was previously enabled.

A cache area is private to each node in a cluster. For a cluster file system, each of the nodes in the cluster has its own cache area. Caching occurs on a per-node basis and the cache contents are not shared across the nodes in the cluster. A file system with caching enabled is associated with the local cache area on each node.

Setting up SmartIO read caching for VxFS file systems

In read mode, the SmartIO feature caches the VxFS file system read I/Os. To set up SmartIO for read caching for a VxFS file system, simply create the cache area.

Setting up SmartIO read caching for VxFS file systems

1. Create the VxFS cache area on the SSD device, using one of the following commands.

   - Specify one or more devices using the disk access name (*daname*) of the device. The device should be initialized for VxVM use and have the `cdsdisk` format.

     ```
     # sfcache create [-t VxFS] [size] daname[...] [--auto|--noauto] \
      [--nostripe|ncols=n] [cachearea_name]
     ```

     Where:
     daname specifies the disk access name of the device on which to create the cache area.
     cachearea_name specifies a customized name for the cache area. If unspecified, the SmartIO feature automatically generates a name for the cache area.
     size specifies the size for the cache area. By default, the cache area uses all of the available space on the device.
     `--nostripe|ncols=`*n* specifies the layout options for the cache area. By default, the cache area is created over a striped volume if two or more disks are specified. Use the `ncols=`*n* option to specify the number of columns for the striped volume. Use the `--nostripe` option to create the cache area on a concatenated volume over the specified disks.
     `--noauto|--auto` specifies the type of cache area. The default is `--auto`.
     For example:

     ```
     # sfcache create ssd0_0
     ```

   - Alternatively, you can specify the name of a disk group and a volume to use for the cache area. In this case, you cannot specify a size: the cache area uses the entire volume.

     ```
     # sfcache create [-t VxFS] [--noauto|--auto] dg/vol
     ```

     Where:
     dg/*vol* specifies the disk group name and volume name that you want to use for the cache area. SmartIO uses this name for the cache area.
     `--noauto|--auto` specifies the type of cache area. The default is `--auto`.

For example:

```
# sfcache create --auto ssd_dg/ssd_vol
```

2 If the file system is not already mounted, mount the VxFS file system.

- If the cache area is auto, read caching is enabled when you mount the VxFS file system.
  For example, for a local mount:

  ```
  # mount -t vxfs /dev/vx/dsk/testdg/vol1 /mnt1
  ```

  For example, for a CFS mount:

  ```
  # mount -t vxfs -o cluster /dev/vx/dsk/testdg/vol1 /mnt1
  ```

- If the cache area is noauto, you must enable caching for each VxFS file system that you want to cache. To enable caching, mount the file system with the `-o smartiomode` option.
  For example, for a local mount:

  ```
  # mount -t vxfs -o smartiomode=read /dev/vx/dsk/testdg/vol1 /mnt1
  ```

  For example, for a CFS mount:

  ```
  # mount -t vxfs -o cluster,smartiomode=read /dev/vx/dsk/testdg/vol1
  /mnt1
  ```

  You can also enable caching after the file system is mounted.

  ```
  # sfcache enable mount_point
  ```

  Where:
  mount_point is the mount point for the file system.
  For example:

  ```
  # sfcache enable /mnt1
  ```

3 If required, you can further customize the caching behavior.

See "Customizing the caching behavior" on page 26.

Verifying the VxFS cache area and monitoring the caching

After the SmartIO feature is configured, you can verify that the cache area is present and that caching is occurring.

For a VxFS cache area, the `sfcache list` command shows the caching mode for the file or directory. If the mode is not explicitly set, the file or directory inherits the caching mode of the mount point. If the mode is explicitly set for a file or directory, that value persists across remounts. The displayed caching mode may differ from the mode that is enabled for the mount point. The `writeback` mode is not enabled unless the file system is mounted in `writeback` mode. If a file or directory is set to `writeback` mode, but the file system is mounted in another mode, the file or directory inherits the caching mode of the mount point.

To verify and monitor the cache area

1. Use the following command to display information about the cache areas on the system.

```
# sfcache list

NAME           TYPE SIZE  ASSOC-TYPE  STATE  FSTYPE  DEVICE
sfcachearea_2 VxFS 7.00g AUTO        ONLINE default sdg
```

2. Use the following command to display information about a specific cache area.

```
# sfcache list sfcachearea_2
Cachearea: sfcachearea_2
Assoc Type: AUTO
Type: VxFS
Size: 9.96g
State: ONLINE
Layout: CONCAT
FStype: default
Number of Columns: 0

/dev/vx/dsk/sfcache_defaultdg/sfcachearea_2:
FSUUID                        SIZE      MODE       MOUNTPOINT     CACHENAME
a6178a5604a50200577a0000a... 759.0 MB  read       /ora_inst1     sfcachearea_2
0d929856a7d60e003d380000f... 512.0 MB  writeback  /pdb2          sfcachearea_2
90929856b688000081410000b... 50.1 MB   read       /fast_recovery sfcachearea_2
02188a569d7e0700567d00007... 4 KB      nocache    -              sfcachearea_2
31188a5679a90900987d00007... 4 KB      nocache    -              sfcachearea_2
a55b8a56db160a00fa610000f... 4 KB      nocache    -              sfcachearea_2
```

3 To display information about a specific file system:

```
# sfcache list /mnt1

/mnt1:
READ CACHE    WRITEBACK    MODE    PINNED  NAME
   39.0 MB         0 KB    read    yes     /mnt1/dir
   39.0 MB         0 KB    read    yes     /mnt1
```

4 To see statistics on the cache usage, use the following command:

```
# sfcache stat /mnt1
```

```
TYPE: VxFS :
       Cache Name: sfcachearea_1
       Cache Size:       5 GB
Cache Utilization:       4 KB ( 0.00 %)

Read Cache                                  Writeback

Hit Ratio Data Read Data Written  Hit Ratio  Data Written rdcachename  wbcachename

/mnt1:
   0.00 %  0 KB     0 KB          0.00 %     0 KB         sfcachearea_1 sfcachearea_2
```

The output displays statistics for the cached data.

See "Viewing the SmartIO cache statistics " on page 84.

5 Use the following command to display information about the usage of exported SSDs by other nodes in the cluster. Run the command on the node for which you want to see the existing cache areas.

Note: Private cache areas of other nodes are not listed.

```
# sfcache list --all
Hostname : sys1
NAME              TYPE    SIZE       ASSOC-TYPE  STATE  FSTYPE    DEVICE
sfcachearea_1     VxVM    10.00g     AUTO        ONLINE   -       ibm_f90-0_0
sfcachearea_3     VxFS    10.00g     AUTO        ONLINE reserve  ibm_f90-0_0

Hostname : sys2
NAME              TYPE    SIZE       ASSOC-TYPE  STATE   FSTYPE   DEVICE
sfcachearea_4     VxFS    20.00g     AUTO        ONLINE  reserve    ibm_f90-0_0
sfcachearea_5     VxVM    25.00g     AUTO        ONLINE     -     ibm_f90-0_0

Hostname : sys3
NAME              TYPE    SIZE       ASSOC-TYPE  STATE   FSTYPE   DEVICE
sfcachearea_2     VxFS    10.00g     AUTO        ONLINE  reserve  ibm_f90-0_0
```

Customizing the caching behavior

By default, SmartIO caches the file data based on the workload. SmartIO loads portions of files into the cache based on I/O access. When the cache area fills, data may be evicted to make room for caching new data. SmartIO uses criteria such as frequency of access to evict data. While the data is in the cache, the subsequent I/Os to that file data are satisfied from the cache. If the data is evicted, any subsequent I/O request is served from the primary storage. SmartIO may then cache the data again.

To maximize the use of the cache, you can customize the caching behavior to control when files are loaded or evicted from the cache. You can customize the caching behavior, using the following operations:

- The `load` operation preloads files into the cache before the I/O accesses the files. The files are already in the cache so that the I/Os return more quickly. By default, the files are loaded in the background. Use the `-o sync` operation to load the files synchronously, which specifies that the command does not return until all the files are loaded. The files that are loaded in this way are subject to the usual eviction criteria.
- The `pin` operation prevents the files from being evicted from the cache. You can pin commonly used files so that SmartIO does not evict the files and later need to cache the files again. A pinned file is kept in the cache indefinitely, until it is deleted or explicitly unpinned. If you pin a file with the `-o load` option, the operation also caches the file contents synchronously. If you do not specify the `-o load` option, the file contents are cached based on I/O access.
- The `unpin` operation removes files from the pinned state. The `unpin` operation does not cause the file to be immediately evicted. SmartIO considers the file for eviction in the same way as any other file, when space is required in the cache.

For each of these operations, you can specify files individually, or specify a directory name to affect all of the files in a directory. Use the `-r` option to make the selection recursive.

To load files or directories

- To load files or directories to the cache, specify one or more file names or directory names to the following command.

  ```
  # sfcache load [-r] [-o sync] {file|dir}[file2|dir2...]
  ```

  Use the `-r` option to make the selection recursive.

  Use the `-o sync` option to specify that the command does not return until all the files are loaded.

To pin files or directories

- To pin files or directories to the cache, specify one or more file names or directory names to the following command.

  ```
  # sfcache pin [-o load] [-r] {file|dir}[file2|dir2...]
  ```

  Use the `-r` option to make the selection recursive.

  Use the `-o load` option to load the file synchronously into the cache.

To unpin files or directories

- To unpin files or directories from the cache, specify one or more file names or directory names to the following command.

  ```
  # sfcache unpin [-r] {file|dir} [file2|dir2...]
  ```

  Use the `-r` option to make the selection recursive.

About SmartIO caching on SSD devices exported by FSS

SmartIO supports the use of Solid-State Drives (SSD) exported by FSS to provide caching services for applications running on Veritas Volume Manager (VxVM) and Veritas File System (VxFS). In this scenario, Flexible Storage Sharing (FSS) exports SSDs from nodes that have a local SSD. FSS then creates a pool of the exported SSDs in the cluster. From this shared pool, a cache area is created for any or all nodes in the cluster. Each cache area is accessible only to that particular node for which it is created. The cache area can be a VxVM cache area or a VxFS cache area.

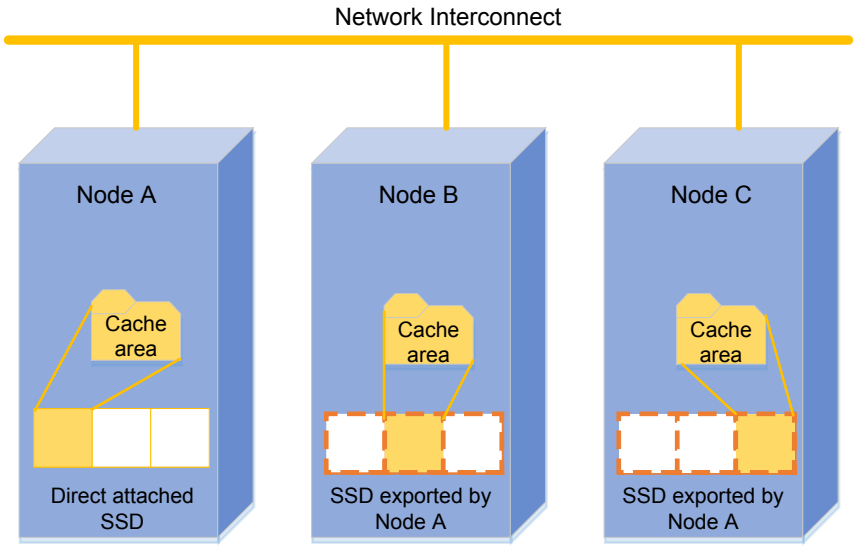
SmartIO supports write-back caching for local mounts on remote SSD devices exported by FSS. However, write-back caching is not supported on remote SSD devices for CFS environments.

If you plan to use only a portion of an exported SSD device for caching purposes, ensure that the volume used for caching is created on a disk group with disk group version 200 or later.

The volume layout of the cache area on remote SSDs follows the simple stripe layout, not the default FSS allocation policy of mirroring across hosts. If the cache area on a remote SSD needs to be resized to meet growing needs, ensure that you specify an exported device only. The operation fails if a non-exported device is specified. The cache areas can be enabled to support warm or persistent caching across reboots.

Figure 2-1 illustrates the caching configuration.

Figure 2-1 SmartIO caching with FSS-exported SSDs



Status of cache areas when nodes leave or join the cluster

Table 2-1 describes the status of the cache when nodes leave or join the cluster.

Table 2-1 Status of cache areas when nodes leave or join the cluster

Scenario	Cache status
Node that uses a remote cache leaves the cluster	Caching is disabled on the node leaving the cluster as the cache area will not be accessible.
Node without a local SSD joins the cluster	When the node joins the cluster, VxVM brings the cache-area online. Caching is enabled for the volumes depending on the association type of the cache area. If the association type is set to `auto`, automatic caching is enabled. If the association type is set to `noauto`, automatic caching is disabled.

Table 2-1 Status of cache areas when nodes leave or join the cluster *(continued)*

Scenario	Cache status
Node that provides storage leaves the cluster	All cache areas created using disks from the node are detached and inaccessible. The remaining nodes in the cluster cannot access the disks. Caching is restarted when the node joins the cluster again.
Node with a local SSD joins the cluster	All cache areas created from the storage of the node are brought online when the node joins the cluster.

Setting up cache areas using SSDs exported by FSS

Log in to the node from which you want to export the SSD device. Then, create the cache area using the exported SSD.

To set up cache areas using SSDs exported by FSS

1. Log in to the node from which you want to export the SSD.
2. Initialize the disk for use with VxVM:

```
# vxdisk init disk_name
```

3. Export the SSD device from the host:

```
# vxdisk export disk_name disk_name
```

4. Log in to the node for which you want to create a cache area using the device exported in step 3.
5. Create the VxVM or VxFS cache area using the exported SSD:

```
# sfcache create [-t cache_type] [cachearea_name]] \
{ssd_device} [size]
```

6. To view the cache areas:

```
# sfcache list --all
```

About SmartIO write-back caching for applications running on VxFS file systems- DLV 11 to 13

InfoScale Storage and InfoScale Enterprise supports write-back caching on solid-state drives (SSDs) for applications running on Veritas File System (VxFS) file systems. In this scenario, application reads and writes are satisfied from the cache whenever possible.

Note: SmartIO write-back caching is not currently supported in SF Oracle RAC environments.

SmartIO provides write caching in the `writeback` mode. In `writeback` mode, an application write returns success after the data is written to the SmartIO cache, which is usually on an SSD. At a later time, SmartIO flushes the cache, which writes the dirty data to the disk. Write-back caching expects to improve the latencies of synchronous user data writes. Write order fidelity is not guaranteed while flushing the dirty data to the disk.

Write-back caching is superset of read caching. When write-back caching is enabled, read caching is implicitly enabled. Reads are satisfied from the cache if possible, and the file system transparently loads file data into the cache. Both read and write-back caching may be enabled for the same file at the same time.

The `writeback` caching mode gives good performance for writes, but also means that the disk copy may not always be up to date. If a cache device fails, a file that is cached in `writeback` mode may not be completely present on the disk. SmartIO has a mechanism to flush the data from the cache device when the device comes back online. Storage Foundation Cluster File System High Availability (SFCFSHA) provides additional protection from data loss with cache reflection.

In the case of SFCFSHA, when `writeback` mode caching is enabled, SmartIO mirrors the write-back data at the file system level to the other node's SSD cache. This behavior, called cache reflection, prevents loss of write-back data if a node fails. If a node fails, the other node flushes the mirrored dirty data of the lost node as part of reconfiguration. Cache reflection ensures that write-back data is not lost even if a node fails with pending dirty data.

In the case of local mount, if there is no cache mirror, the disk copy of the file can be incomplete or stale if the node fails.

After write-back caching is enabled on the mount point, the qualified synchronous writes in that file system are cached. SmartIO determines if a write qualifies for write-back caching, using criteria such as the following:

- The write request must be PAGESIZE aligned (multiple of 4k).

- The write request is not greater than 2MB.
- The file on which the writes are happening is not mmapped
- The `writeback` mode caching is not explicitly disabled by the administrator.

You can also customize which data is cached, by adding advisory information to assist the SmartIO feature in making those determinations.

Required configuration for SmartIO write-back caching for a VxFS file system

You must have an Enterprise license to use SmartIO with write-back caching for Storage Foundation or Storage Foundation High Availability.

You can set up SmartIO for write-back caching for a VxFS file system with the following configurations:

- A Storage Foundation Cluster File System High Availability (SFCFSHA) cluster with exactly 2 nodes. Write-back caching is not enabled if the cluster has more than 2 nodes. If another node is added while `writeback` mode caching is configured, write-back caching is disabled. Caching continues in read mode. If the cluster file system is unmounted on one of the two nodes while `writeback` mode caching is configured, then write-back caching is disabled. Caching continues in read mode. If the cluster file system on the second node is remounted, then write-back caching is enabled automatically.
- Local mount configuration.

In the case of CFS, write-back caching uses LLT transport to mirror the write-back data. Application writes that are cached are also written to the remote cache before the write is returned.

Veritas recommends that you configure LLT over a high bandwidth network such as 10GigE or RDMA to avoid impact to the throughput.

For information on configuring LLT, see the *Storage Foundation Cluster File System High Availability Configuration and Upgrade Guide*.

The file system must be mounted on a VxVM volume.

Setting up SmartIO write-back caching for VxFS file systems

In `writeback` mode, the SmartIO feature caches the VxFS file system read and write I/Os. To set up SmartIO for write-back caching for a VxFS file system, create the cache area and mount the file system in `writeback` mode.

Setting up SmartIO write-back caching for VxFS file systems

1. Create the VxFS cache area on the SSD device, using one of the following commands.

   - Specify one or more devices using the disk access name (*daname*) of the device. The device should be initialized for VxVM use and have the `cdsdisk` format.

     ```
     # sfcache create [-t VxFS] [size] daname[...] [--auto|--noauto] \
      [--nostripe|ncols=N] [cachearea_name]
     ```

     Where:
     daname specifies the disk access name of the device on which to create the cache area.
     cachearea_name specifies a customized name for the cache area. If unspecified, the SmartIO feature automatically generates a name for the cache area.
     size specifies the size for the cache area. By default, the cache area uses all of the available space on the device.
     `--noauto|--auto` specifies the type of cache area. The default is `--auto`.
     For example:

     ```
     # sfcache create ssd0_0
     ```

     `--nostripe|ncols=`*n* specifies the layout options for the cache area. By default, the cache area is created over a striped volume if two or more disks are specified. Use the `ncols=`*n* option to specify the number of columns for the striped volume. Use the `--nostripe` option to create the cache area on a concatenated volume over the specified disks.

   - Alternatively, you can specify the name of a disk group and a volume to use for the cache area. In this case, you cannot specify a size: the cache area uses the entire volume.

     ```
     # sfcache create [-t VxFS] [--noauto|--auto] dg/vol
     ```

     Where:
     `--noauto|--auto` specifies the type of cache area. The default is `--auto`.
     dg/*vol* specifies the disk group name and volume name that you want to use for the cache area. SmartIO uses this name for the cache area.
     For example:

```
# sfcache create --auto ssd_dg/ssd_vol
```

2 Mount the VxFS file system and set the `smartiomode` option to `writeback`. If the file system is already mounted, you must remount the file system to set the `smartiomode` option to `writeback`.

Example of local mount:

```
# mount -t vxfs -o smartiomode=writeback /dev/vx/dsk/testdg/vol1 \
 /mnt1
```

Example of remount:

```
# mount -t vxfs -o remount,smartiomode=writeback \
/dev/vx/dsk/testdg/vol1 /mnt1
```

For a cluster file system, the file system must be mounted on both the nodes with the `smartiomode` and `cluster` options set.

When you mount the CFS file system with these options, SmartIO automatically reflects the cache on the other node's SSD cache.

Example of CFS mount:

```
# mount -t vxfs -o cluster,smartiomode=writeback \
/dev/vx/dsk/testdg/vol1 /mnt1
```

To enable write-back caching, the `smartiomode` option must be set to `writeback` regardless of whether the cache area is `auto` or `noauto`. If the cache area is `auto` and the `smartiomode` is not set, SmartIO caching is enabled in the default read mode for the file system.

3 If required, you can further customize the caching behavior.

See "Customizing the caching behavior" on page 26.

See "Tuning the writeback caching" on page 82.

Verifying the VxFS cache area and monitoring the caching (writeback mode)

After the SmartIO feature is configured, you can verify that the cache area is present and that caching is occurring.

For a VxFS cache area, the `sfcache list` command shows the caching mode for the file or directory. If the mode is not explicitly set, the file or directory inherits the caching mode of the mount point. If the mode is explicitly set for a file or directory, that value persists across remounts. The displayed caching mode may differ from

the mode that is enabled for the mount point. The `writeback` mode is not enabled unless the file system is mounted in `writeback` mode. If a file or directory is set to `writeback` mode, but the file system is mounted in another mode, the file or directory inherits the caching mode of the mount point.

To verify and monitor the cache area

1. To display information about the cache areas on the system.

   ```
   # sfcache list
   ```

   For example, a single-node VxFS cache area displays output as follows:

   ```
   NAME          TYPE SIZE  ASSOC-TYPE  STATE  FSTYPE  DEVICE
   sfcachearea_2 VxFS 7.00g AUTO        ONLINE default sdg
   ```

   For a cluster file system with `writeback` enabled, if you issue the `sfcache list` command just after you mount with `writeback` enabled (and before any read caching has happened), the output shows the space used in cache as 1.0 GB. Cache reflection is configured, with a local log and a remote log that each have a log size of 512 MB.

2. To display information about a specific cache area:

   ```
   # sfcache list sfcachearea_2
   ```

```
Cachearea: sfcachearea_2
Assoc Type: AUTO
Type: VxFS
Size: 9.96g
State: ONLINE
Layout: CONCAT
FStype: default
Number of Columns: 0

/dev/vx/dsk/sfcache_defaultdg/sfcachearea_2:
FSUUID                        SIZE      MODE       MOUNTPOINT     CACHENAME
a6178a5604a50200577a0000a... 759.0 MB  read       /ora_inst1     sfcachearea_2
0d929856a7d60e003d380000f... 512.0 MB  writeback  /pdb2          sfcachearea_2
90929856b688000081410000b... 50.1 MB   read       /fast_recovery sfcachearea_2
02188a569d7e0700567d00007... 4 KB      nocache    -              sfcachearea_2
31188a5679a90900987d00007... 4 KB      nocache    -              sfcachearea_2
a55b8a56db160a00fa610000f... 4 KB      nocache    -              sfcachearea_2
```

The output displays information about the cache configuration and attributes.

See “Displaying information about a cache area” on page 73.

3 To display information about a specific file system:

```
# sfcache list /mnt1

/mnt1:
READ CACHE    WRITEBACK    MODE        PINNED  NAME
   39.0 MB         0 KB    writeback   yes     /mnt1/dir
   39.0 MB         0 KB    writeback   yes     /mnt1
```

4 To see statistics on the cache usage, use the following command:

```
# sfcache stat sfcachearea_1
```

```
TYPE: VxFS
NAME: sfcachearea_1
       Cache Name: sfcachearea_1
              Cache Size:       5 GB
        Cache Utilization:   1.426 GB (28.51 %)
File Systems Using Cache:       2
Writeback Cache Use Limit:  Unlimited
  Writeback Flush Timelag:      10 s

Read Cache                                  Writeback
Hit Ratio  Data Read  Data Written    Hit Ratio  Data Written rdcachename  wbcachename

Total:
   7.98 %  157.7 MB   1.545 GB       0.00 %      0 KB


/pdb1:
   7.98 %  157.7 MB   1.545 GB       0.00 %      0 KB        sfcachearea_1     -


/pdb2:
   0.00 %  0 KB       0 KB           0.00 %      0 KB        sfcachearea_1  sfcachearea_2
```

The output displays statistics for the cached data.

See “Viewing the SmartIO cache statistics ” on page 84.

5 To see statistics on cache usage for a particular file system, use the following command:

```
# sfcache stat /mnt1
```

```
TYPE: VxFS :
       Cache Name: sfcachearea_1
       Cache Size:       5 GB
Cache Utilization:       4 KB ( 0.00 %)

Read Cache                                  Writeback

Hit Ratio Data Read Data Written  Hit Ratio  Data Written rdcachename  wbcachename

/mnt1:
   0.00 %  0 KB     0 KB          0.00 %     0 KB         sfcachearea_1 sfcachearea_2
```

6 Check the `syslog` to verify whether `writeback` mode caching is enabled.

You should see a line such as the following in the `syslog`:

```
vxfs: msgcnt 4 writeback caching is enabled for /dev/vx/dsk/testdg/vol1
```

If `writeback` mode caching is disabled for a particular file system, you would see a line such as the following in the `syslog`:

```
vxfs: msgcnt 9 writeback caching is disabled for /dev/vx/dsk/testdg/vol1
```

About SmartIO FEL-based writeback caching for applications running on VxFS file systems- DLV 14 and later

InfoScale Storage and InfoScale Enterprise supports write-back caching on solid-state drives (SSDs) for applications running on Veritas File System (VxFS) file systems. In case of DLV 14 and later, application writes are cached in a Front-End Log (FEL)-based writeback device.

The FEL-based writeback caching mechanism is functionally similar to that of the SmartIO write-back cache. However, many aspects of the FEL-based design is expected to result in better performance than SmartIO write-back caching. Also, the SmartIO FEL-based write-back caching removes the restriction of two node cluster, which was present in the SmartIO write-back cache. Since both the caching types are functionally similar, the FEL based write-back caching is a replacement

for the SmartIO write-back caching, on DLV 14 and later. FEL-based write-back cache continues to use many of the user management interfaces as the SmartIO write-back cache. For older disk layouts i.e. DLV 13 and prior, SmartIO write-back caching mechanism remains unchanged.

An FEL-based write-back cache is a volume that is separate from the primary volumes of a filesystem which is used as a persistent staging area, allowing synchronous writes to return quickly to the user, regardless of the complexity of the write itself. Writes to VxFS can potentially require a lot of processing, depending on the features in use, before they are written persistently to disk. For example, write to a sparse region of a file requires allocation in the context of the write itself, which can introduce significant delay to the synchronous write. It is difficult for a user or application to predict this delay ahead of time. The FEL-based write-back caching helps to buffer this variability, providing minimal and predictable write latency.

Note: SmartIO FEL-based write-back caching is not currently supported in SF Oracle RAC environments.

In `writeback` mode, an application write returns success after the data is written to the cache, which is usually on an SSD. At a later time, data is flushed to the primary storage. Write order fidelity is not guaranteed while flushing the dirty data to the disk.

The `writeback` caching mode gives good performance for writes, but also means that the disk copy may not always be up to date. Because data from successful writes might only exist persistently in FEL, until it is flushed, availability of the FEL volume is as important as that of the main filesystem volumes. If a cache device fails, a file that is cached in `writeback` mode may not be completely present on the disk. SmartIO has a mechanism to flush the data from the cache device when the device comes back online.

Storage Foundation Cluster File System High Availability (SFCFSHA) provides additional protection from data loss by mirroring the FEL cache between nodes using FSS. The filesystem can recover from a node failure by reading the FEL from a mirror on another node, resulting in full access to the filesystem without requiring the failed node to come back.

In case of SFCFSHA, when `writeback` caching mode is enabled, SmartIO mirrors the write-back data at the file system-level, to the other node's SSD cache. This behavior, called cache reflection, prevents loss of write-back data if a node fails. In case of a node failure, the other node flushes the mirrored dirty data of the lost node as part of reconfiguration. Cache reflection ensures that the write-back data is not lost even if a node fails with pending dirty data.

In the case of local mount, if there is no cache mirror, the disk copy of the file can be incomplete or stale if the FEL cache fails.

After the FEL-based write-back caching is enabled on the mount point, the qualified synchronous writes in that file system are cached. SmartIO determines if a write qualifies for write-back caching, using criteria such as the following:

- The write request must be PAGESIZE aligned (multiple of 4k).
- The write request is not greater than 2MB.
- The file on which the writes are happening is not memory-mapped.
- The FEL-based `writeback` mode caching is not explicitly disabled by the administrator.

Required configuration for FEL-based write-back caching for a VxFS file system

You must have an Enterprise license to use SmartIO FEL based write-back caching for InfoScale Storage or InfoScale Enterprise

You can set up SmartIO for FEL-based write-back caching for a VxFS file system with the following:

- A Storage Foundation Cluster File System High Availability (SFCFSHA) cluster configuration
- Local mount configuration

In the case of CFS, FEL cache must be created on SSD devices exported by FSS.

Setting up FEL-based write-back caching for VxFS file systems

To set up FEL-based write-back caching for a VxFS file system, create an FEL cache area and mount the file system in `writeback` mode.

Setting up FEL-based write-back caching for VxFS file systems

1. Create an FEL cache area on an SSD device, using one of the following commands.

   Specify one or more devices using the disk access name (daname) of the device. The device should be initialized for VxVM use and should have the cdsdisk format.

   - `# sfcache create [`*`size`*`] daname[...] [ --fel [dgname=name] [nmirrors=N] [node1:`*`daname node2`*`:`*`daname`*`]] [`*`cachearea_name`*`]`

     Where:

daname specifies the disk access name of a device on which you want to create a cache area.

cachearea_name specifies a customized name for an FEL cache area. If unspecified, SmartIO automatically generates a name.

size specifies the size for a cache area. By default, the cache area uses all of the available space on the device.

dgname specifies the name of an already created FSS disk group.

nmirrors specifies the number of mirrored volumes to be created on FSS diskgroup.

node1,*node2* specifies the cluster node names.

For example:

```
# sfcache create --fel ssd0_0
```

In SFCFSHA configuration:

```
# sfcache create --fel node1:ssd0_0 node2:ssd1_0
```

- Alternatively, specify the name of an FSS disk group and a volume to use for an FEL cache area. In this case, you cannot specify a size. The FEL cache area uses the entire volume.

  For example:

  ```
  # sfcache create [--fel] dg/vol
  ```

  Where:

  dg/vol is the FSS disk group name and volume name. SmartIO uses this name for the FEL cache area.

  For example:

  ```
  # sfcache create --fel ssd_dg/ssd_vol
  ```

2 Bring the FEL cache online on each node.

```
sfcache online CacheName.
```

3 Mount the VxFS file system and set the smartiomode option to `writeback`. If the file system is already mounted, you must remount the file system to set the smartiomode option to `writeback`.

For local mount:

```
# mount -t vxfs -o smartiomode=writeback /dev/vx/dsk/testdg/vol1
/mnt1
```

For remount:

```
# mount -t vxfs -o remount,smartiomode=writeback
/dev/vx/dsk/testdg/vol1 /mnt1
```

For a cluster file system, the file system must be mounted on all the nodes with the smartiomode and cluster options set.

For CFS mount:

```
# mount -t vxfs -o cluster,smartiomode=writeback
/dev/vx/dsk/testdg/vol1 /mnt1
```

To enable writeback caching, the smartiomode option must be set to `writeback`.

4 Customize the caching behavior further, if required.

See "Customizing the caching behavior" on page 26.

See "Tuning the writeback caching" on page 82.

Verifying the VxFS cache area and monitoring FEL-based write-back caching

After the SmartIO feature is configured, you can verify that the FEL cache area is present and that caching is occurring.

For an FEL cache area, the `sfcache list` command shows the caching mode for the file or directory. If the mode is not explicitly set, the file or directory inherits the caching mode of the mount point. If the mode is explicitly set for a file or directory, that value persists across remounts. The displayed caching mode may differ from the mode that is enabled for the mount point. The `writeback` mode is not enabled unless the file system is mounted in `writeback` mode. If a file or directory is set to `writeback` mode, but the file system is mounted in another mode, the file or directory inherits the caching mode of the mount point.

To verify and monitor the cache area

1 List the FEL cache areas on a system.

```
# sfcache list
```

For example, a single-node FEL cache area displays output as follows:

```
NAME  TYPE  SIZE  ASSOC-TYPE  STATE  FSTYPE  DEVICE
 feldg/fel_vol   VxFS  59.96g   NONE    ONLIN     fel   sde
```

2 Verify details about a specific cache area.

```
# sfcache list feldg/fel_vol
```

For example, the FEL cache area displays output as follows:

```
Cachearea: feldg/fel_vol
Assoc Type: NONE
Type: VxFS
Size: 59.96g
State: ONLINE
Layout: CONCAT
FStype: fel
Number of Columns: 0


FEL: feldg/fel_vol UUID: 50e8ef5a60e50a00227d000056276c6c
Chunk-size   Total-chunks   Free-chunks  Available     Avail%
   4194304        15351       15346       59.9 GB      99.97%
FS-id UUID                                    Chunks Use%
    1 d3e8ef5a8e3002001b0600007d92750e          1    0.00651%
    2 d4e8ef5af2320400210600003e1be94e          1    0.00651%
```

3 Verify details about a specific file system.

```
# sfcache list /mnt1
```

For example, the FEL cache area displays output as follows:

```
/mnt1:
READ CACHE    WRITEBACK    MODE          PINNED      NAME
  0 KB         97.5 MB    writeback      no       /mnt1/dir
  0 KB          0 KB      writeback      no       /mnt1/file_2
  0 KB         97.5 MB    writeback      no       /mnt1
```

4 Check the statistics on cache usage:

```
# sfcache stat sfcachearea_1
```

See “Viewing the SmartIO cache statistics ” on page 84.

5 Check the FEL cache usage for a particular file system.

```
# sfcache stat /mnt1
```

About multiple SmartIO cache areas for read and write-back caching on VxFS file systems

You can create multiple cache areas on single and multi-node systems running the VxFS file system. The cache areas can be different for read and write-back operations. Each application can use one cache area for its read operation and another cache area for write-back operations. You can configure the same cache area to be used as a read cache for some applications as well as a write-back cache for other applications.

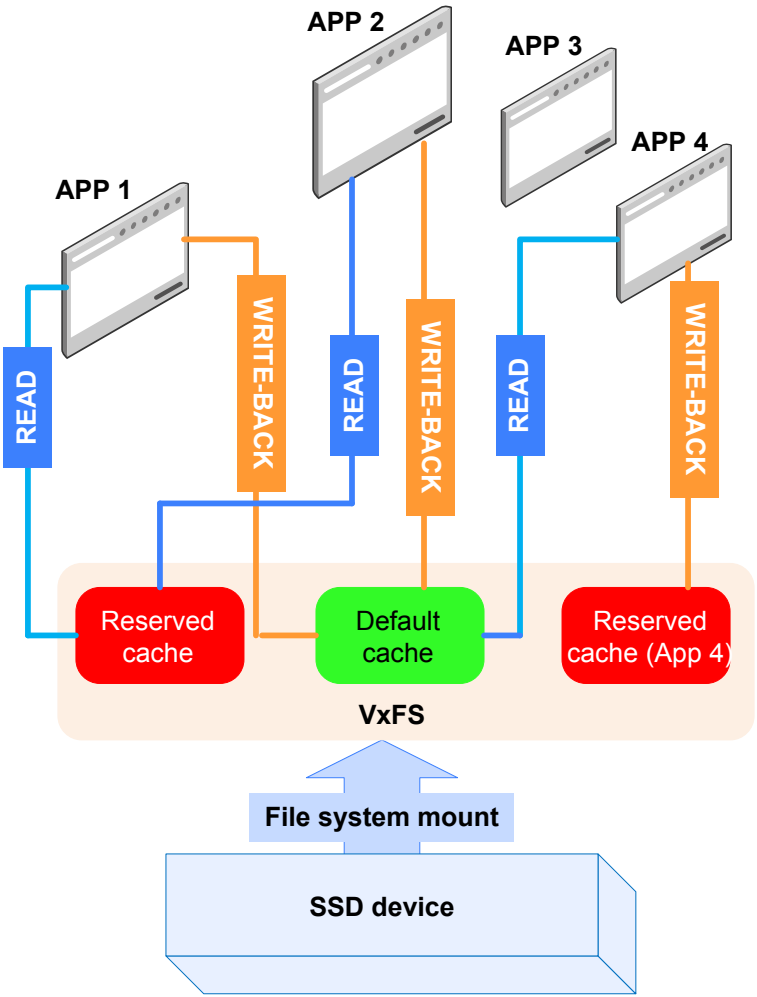
VxFS supports the following types of cache areas:

Default	Available to all applications except those that are configured to use a reserved cache area. You can have only one default cache area online at a time. It is not mandatory to have a default cache area.
Reserved	Reserved for use by certain applications. Available only to applications that are configured to use the cache area. One or more applications may use the same cache area as a reserved cache area. You can have any number of reserved cache areas.
Front-End Log (FEL)	Needed for FEL-based writeback caching from DLV 14 onwards. Multiple FEL cache areas are not yet supported.

You can set the cache type at the time of creating the cache area. The cache area is set to "default" if not specified otherwise. Cache areas can be converted from one type to another. If a cache area is not defined for an application, the default cache area is used.

Figure 2-2 illustrates read and write-back caching on multiple SmartIO cache areas.

Figure 2-2 Read and write-back caching on multiple SmartIO cache areas



1. You can have only 1 default cache area.

2. You can have any number of reserved cache areas.

3. A reserved cache area can be used by multiple applications.

Figure 2-3 illustrates the steps to create multiple SmartIO cache areas on a single node system.

Figure 2-3 Steps to create multiple SmartIO cache areas on a single node system

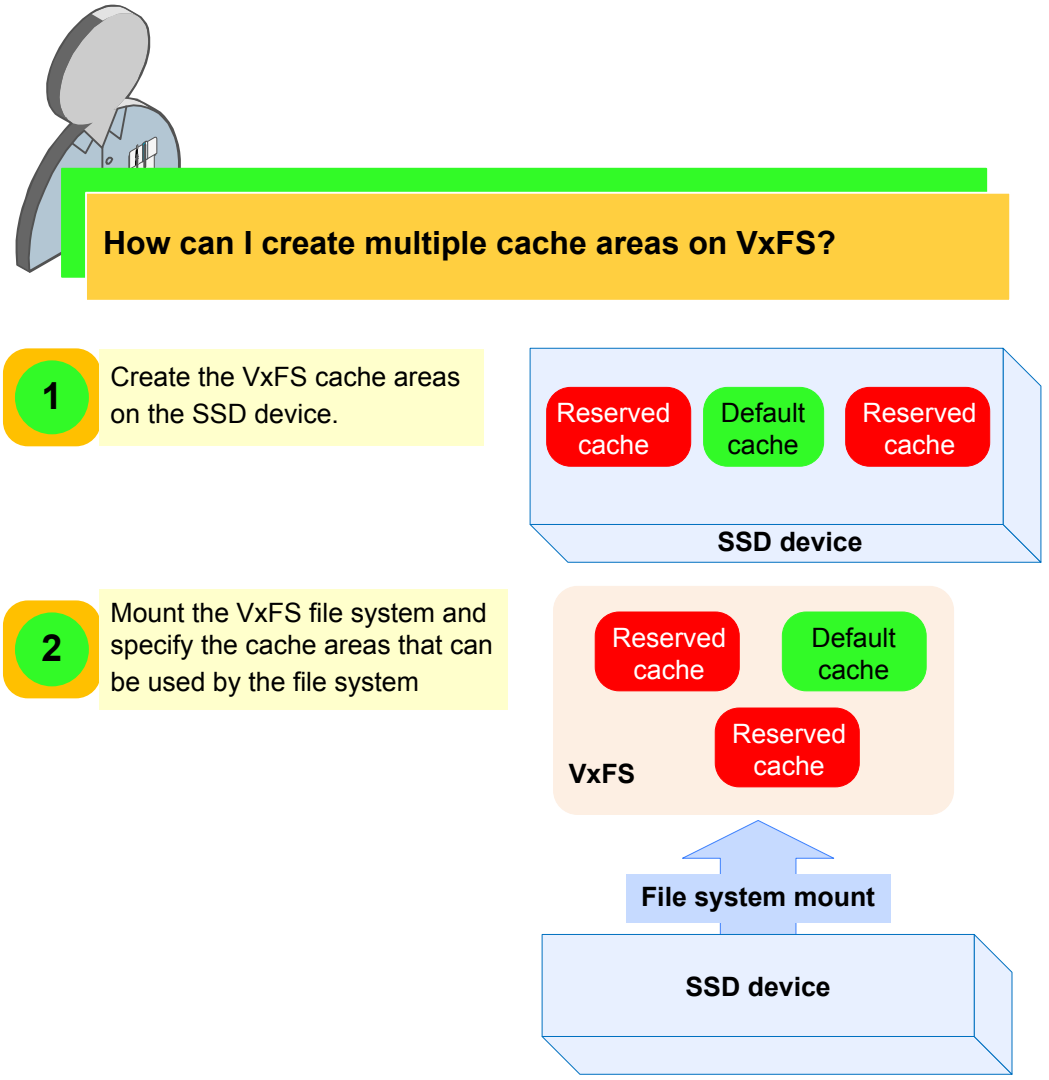
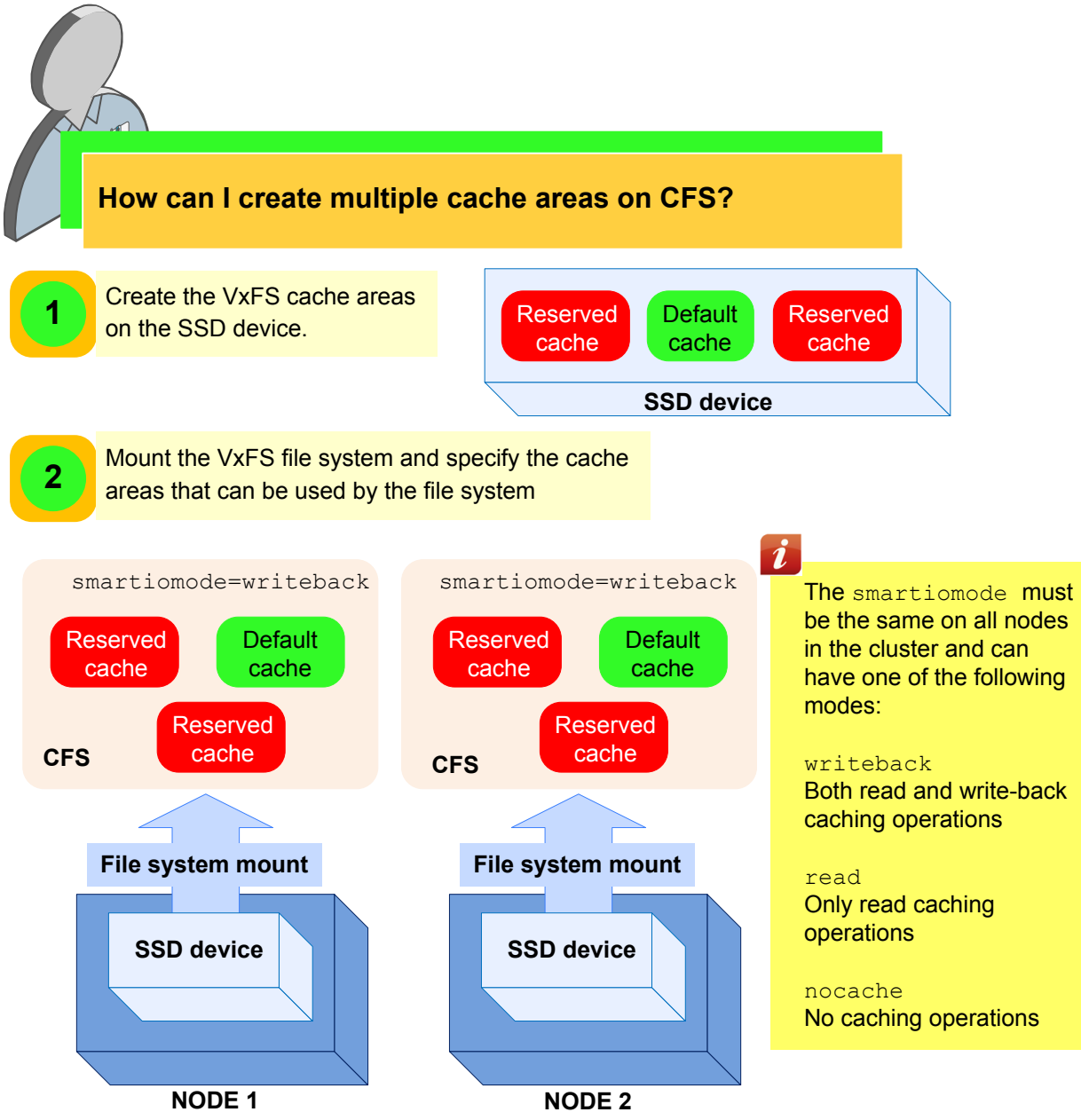


Figure 2-4 illustrates the steps to create multiple SmartIO cache areas on a multi-node cluster.

Figure 2-4 Steps to create multiple SmartIO cache areas on a multi-node cluster



About the `smartiocache` option

The `smartiocache` option of the `mount` command lets you specify the cache areas that can be used by a file system. You can specify different cache areas for read and writeback caching on a file system.

```
mount -t vxfs -o smartiomode=[read|writeback],smartiocache=[cachearea_name]
\ [[rdcachearea_name]:[wbcachearea_name]] file_system \mount_point
```

Where:

`cachearea_name` specifies the name of the SmartIO cache area.

`rdcachearea_name` specifies the name of the cache area for read caching

`wbcachearea_name` specifies the name of the cache area for write-back caching

- The `smartiocache` values depend on the value specified for `smartiomode`
- You can change the cache area name when you remount the file system. When a new cache area is specified, the existing cache area is detached from the file system.
  If the specified cache area is not online at the time of mounting the file system, then the file system cannot start using the cache area. The cache area can be used by the file system after it comes online.
- If only one cache area name is specified and the `smartiomode` is set to `writeback`, the cache area is used for both read and write-back caching operations.
  If no cache area name is specified and the default cache area is online in auto mode, then the default cache area is used for caching based on the `smartiomode` setting.
- If the read cache area is brought offline for a mount point, then write-back caching is also disabled for the mount point. It will be enabled again when read caching is enabled.

Converting VxFS cache areas from one type to another

You can switch a VxFS cache area between reserved and default states whenever the cache is online.

Cache conversions fail in the following scenarios:

- Cache is offline.
- Conversion of a reserved cache area to default, if a default cache area is already present. VxFS supports only one online default cache area on a system.

For example, if the cache area `cache1` is of type reserved and you want to make the cache area available as a default cache area, run the following command:

```
# sfcache set type=default cache1
```

If the cache area `cache2` is of type default and you want to reserve the cache area for a particular application, run the following command:

```
# sfcache set type=reserve cache2
```

Setting up multiple cache areas on a system

First, create the cache areas. Then, mount the file system and specify the cache areas that can be used by the file system.

To set up multiple cache areas on a system

1. Create the VxFS cache area on the SSD device, using one of the following commands.

   - Specify one or more devices using the disk access name (*daname*) of the device. The device should be initialized for VxVM use and have the `cdsdisk` format.

     ```
     # sfcache create [-t VxFS] [size]
     daname[...] [--auto|--noauto] \
      [--default|--reserve] [cachearea_name]
     ```

     Where:
     daname specifies the disk access name of the device on which to create the cache area.
     cachearea_name specifies a customized name for the cache area. If unspecified, the SmartIO feature automatically generates a name for the cache area.
     size specifies the size for the cache area. By default, the cache area uses all of the available space on the device.
     `--noauto|--auto` specifies the cache area mode. The default is `--auto`.
     `--default|--reserve` specifies the type of cache area. The default is `--default`
     For example:

     ```
     # sfcache create ssd1 --reserve cache1
     ```

   - Alternatively, you can specify the name of a disk group and a volume to use for the cache area. In this case, you cannot specify a size; the cache area uses the entire volume.

```
# sfcache create [-t VxFS] [--noauto|--auto] [--default|--reserve]
 dg/vol
```

Where:

`--noauto|--auto` specifies the cache area mode. The default is `--auto`.

`--default|--reserve` specifies the type of cache area. The default is `--default`

dg/*vol* specifies the disk group name and volume name that you want to use for the cache area. SmartIO uses this name for the cache area.

For example:

```
# sfcache create --auto --reserve ssd_dg/ssd_vol
```

2 Mount the VxFS file system and specify the cache areas that can be used by the file system.

In the following example, *cache1* is used for read caching and *cache2* is used for write-back caching:

```
# mount -t vxfs -o smartiomode=writeback,smartiocache=cache1:cache2 \
/dev/vx/dsk/testdg/vol1 \
 /mnt1
```

In the following example, *cache1* is used for read caching:

```
# mount -t vxfs -o smartiomode=read,smartiocache=cache1 \
/dev/vx/dsk/testdg/vol1 /mnt1
```

In the following example, *cache1* is used for read caching as well as write-back caching:

```
# mount -t vxfs -o smartiomode=writeback,smartiocache=cache1 \
/dev/vx/dsk/testdg/vol1 /mnt1
```

For a cluster file system, the file system must be mounted on both the nodes with the `cluster` and the `smartiomode` options set. The `smartiomode` must be the same on all nodes in the cluster.

Example of CFS mount:

```
# mount -t vxfs -o cluster,smartiomode=writeback,\
smartiocache=cache1:cache2 /dev/vx/dsk/testdg/vol1 /mnt1
```

If a default cache exists, and no `smartiocache` option is specified, VxFS automatically uses the default cache for the specified caching mode.

```
# mount -t vxfs -o cluster,smartiomode=writeback,\
/dev/vx/dsk/testdg/vol1 /mnt1
```

3 If required, you can further customize the caching behavior.

See “Customizing the caching behavior” on page 26.

See “Tuning the writeback caching” on page 82.

Verifying the VxFS cache areas

After you configure the cache areas, verify that the cache area is present and that caching is occurring.

To verify the VxFS cache areas

1. To display information about the cache areas on the system.

   ```
   # sfcache list
   ```

   For example, a single-node VxFS cache area displays output as follows:

```
NAME            TYPE    SIZE        ASSOC-TYPE   STATE       FSTYPE      DEVICE
sfcachearea_1   VxFS    5.00g       AUTO         ONLINE      reserve     ssd0_0
sfcachearea_2   VxFS    7.00g       AUTO         ONLINE      default     ssd0_1
```

   For a cluster file system with `writeback` enabled, if you issue the `sfcache list` command just after you mount with `writeback` enabled (and before any read caching has happened), the output shows the space used in cache as 1.0 GB. Cache reflection is configured, with a local log and a remote log that each have a log size of 512 MB.

2. To display information about a specific cache area:

   ```
   # sfcache list sfcachearea_2
   ```

```
Cachearea: sfcachearea_2
Assoc Type: AUTO
Type: VxFS
Size: 9.96g
State: ONLINE
Layout: CONCAT
FStype: default
Number of Columns: 0

/dev/vx/dsk/cachedg/ssdvol:
FSUUID                                          SIZE     MODE      MOUNTPOINT CACHENAME
5efe4a52eb76000041760000a0dec33fe70300005efe4a52 39.0MB   read      /mnt2      sfcachearea_2
```

   The output displays information about the cache configuration and attributes.

   See “Displaying information about a cache area” on page 73.

3. To display information about a specific file system:

   ```
   # sfcache list /mnt1
   ```

```
/mnt1:
READ CACHE    WRITEBACK    MODE       PINNED  NAME
   39.0 MB         0 KB    writeback  yes     /mnt1/dir
   39.0 MB         0 KB    writeback  yes     /mnt1
```

4 To see statistics on the cache usage, use the following command:

```
# sfcache stat sfcachearea_1

TYPE: VxFS
NAME: sfcachearea_1
       Cache Name: sfcachearea_1
               Cache Size:       5 GB
        Cache Utilization:   1.426 GB (28.51 %)
File Systems Using Cache:       2
Writeback Cache Use Limit:  Unlimited
  Writeback Flush Timelag:      10 s

Read Cache                                  Writeback
Hit Ratio  Data Read  Data Written Hit Ratio  Data Written  rdcachename

Total:
   7.98 %  157.7 MB   1.545 GB      0.00 %      0 KB

/pdb1:
   7.98 %  157.7 MB   1.545 GB      0.00 %      0 KB       sfcachearea_1

/pdb2:
   0.00 %  0 KB       0 KB          0.00 %      0 KB       sfcachearea_1
```

The output displays statistics for the cached data.

See “Viewing the SmartIO cache statistics ” on page 84.

5 To see statistics on cache usage for a particular file system, use the following command:

```
# sfcache stat /mnt1
```

```
Cache Size:    9.97 GB
Cache Utilization:   551.0 MB ( 5.40 %)

Read Cache                          Writeback
Hit Ratio  Data Read Data Written Hit Ratio Data Written rdcachename wbcachename

/mnt1:
   0.00 %  0 KB      78.0 MB      100.00 %   39.0 MB   sfcachearea_1 sfcachearea_2
```

6 Check the `syslog` to verify whether `writeback` mode caching is enabled.

You should see a line such as the following in the `syslog`:

```
vxfs: msgcnt 4 writeback caching is enabled for /dev/vx/dsk/testdg/vol1
```

If `writeback` mode caching is disabled for a particular file system, you would see a line such as the following in the `syslog`:

```
vxfs: msgcnt 9 writeback caching is disabled for /dev/vx/dsk/testdg/vol1
```

About SmartIO caching for Oracle databases on VxFS file systems

SmartIO provides application templates to optimize caching for databases running on VxFS file systems. SmartIO uses the templates to apply policies to particular types of information in the database. For example, index files may have different caching policies from data files.

SmartIO provides the following application template for VxFS file systems:

- Template name: oracle
  Optimizes caching for Oracle databases running on VxFS file systems.

Prerequisites and configuration for using the SmartIO plug-in for Oracle

The SmartIO feature of Storage Foundation and High Availability Solutions includes a plug-in for Oracle databases.

The SmartIO plug-in for Oracle requires Oracle version 11 or later. This restriction does not apply to SmartIO caching without the plug-in.

Make sure the system is configured correctly to use the SmartIO plug-in for Oracle.

To configure the system to use the SmartIO plug-in for Oracle

1. Before executing the Oracle SmartIO plug-in, create the file system cache area and bring the cache area online.
2. The Oracle SmartIO plug-in needs to query database catalog tables. Make sure the Oracle database is online and running on the same host where you need to run the `sfcache` commands for the Oracle plug-in.
3. For the `sfcache app oracle` command to work, the `/etc/oratab` file must be present and must include the following line:

   oraclesid`:`*oracle_home*`:Y|N:`

   Where:

   oraclesid is the system id (SID) of an Oracle instance on the server.

   oracle_home is the ORACLE_HOME directory associated with this instance.

   `Y|N` flags indicate whether the instance should automatically start at boot time.

Setting default SmartIO caching policies for a database running on a VxFS file system

SmartIO provides application templates to optimize caching for databases running on VxFS file systems. SmartIO uses the templates to apply policies to particular types of files in the database. For example, caching policies for index files and data files.

The oracle template sets the default policy for the Oracle database, as follows:

- Turns off caching (nocache mode) for ARCHLOG files
- Sets read caching (read mode) for TEMPFILEs
- For OLTP database, sets read caching to the datafiles with the most frequent reads.
  For OLAP database, sets read caching to all datafiles that contain INDEXes.

To set the default SmartIO caching policies for a database, run the following command as ORACLE user:

```
# sfcache app [cachearea=cachearea_name] oracle -S $ORACLE_SID \
   -H $ORACLE_HOME -o setdefaults --type={OLTP | OLAP}
```

Where:

$ORACLE_HOME and $ORACLE_SID are mandatory and uniquely identify the database.

OLAP or OLTP indicates the type of application load. OLAP, or Online Analytical Processing, applications process workloads intended for multi-dimensional analytical queries. OLTP, or Online Transaction Processing, applications process transaction-oriented workloads, such as for data entry and retrieval transaction processing.

Example of an OLTP database:

```
$ sfcache app cachearea=sfcachearea_1 oracle -S $ORACLE_SID \
-H $ORACLE_HOME -o setdefaults --type=oltp

INFO: Oracle Instance tpcc is running
INFO: Store DB details at /tpccdata/.CACHE_INFO
INFO: Setting oltp policies
INFO: Setting nocache mode to /tpccdata

INFO: Setting nocache mode to /tpcclog

INFO: Setting nocache mode to /tpcclog/log_1_1
INFO: Setting nocache mode to /tpcclog/log_1_2
INFO: Setting nocache mode to /tpcclog/log_1_3
INFO: Setting nocache mode to /tpcclog/log_1_4

INFO: Setting nocache mode to /home/oracle/app/oracle/product/11.2.0
/dbhome_1/dbs/arch
INFO: Setting read mode to /tpccdata/temp_0_0
INFO: Setting read mode to /tpccdata/temp_0_2
INFO: Setting read mode to /tpccdata/temp_0_1
INFO: Setting read mode to /tpccdata/temp_0_4

AWR snapid range min=1 max=7

INFO: Setting read mode to /tpccdata/stok_0_25
INFO: Setting read mode to /tpccdata/stok_0_24
INFO: Setting read mode to /tpccdata/stok_0_20
INFO: Setting read mode to /tpccdata/stok_0_29
INFO: Setting read mode to /tpccdata/stok_0_23
INFO: Setting read mode to /tpccdata/stok_0_22
INFO: Setting read mode to /tpccdata/cust_0_5
```

Example of an OLAP database:

```
$ sfcache app cachearea=sfcachearea_1 oracle -S $ORACLE_SID \
-H $ORACLE_HOME -o setdefaults --type=olap
INFO: Oracle Instance tpcc is running
INFO: Store DB details at /tpccdata/.CACHE_INFO
INFO: Setting olap policies
INFO: Setting nocache mode to /tpccdata

INFO: Setting nocache mode to /tpcclog

INFO: Setting nocache mode to /tpcclog/log_1_1
INFO: Setting nocache mode to /tpcclog/log_1_2
INFO: Setting nocache mode to /tpcclog/log_1_3
INFO: Setting nocache mode to /tpcclog/log_1_4

INFO: Setting nocache mode to /home/oracle/app/oracle/product/11.2.0
/dbhome_1/dbs/arch
INFO: Setting read mode to /tpccdata/temp_0_0
INFO: Setting read mode to /tpccdata/temp_0_2
INFO: Setting read mode to /tpccdata/temp_0_1
INFO: Setting read mode to /tpccdata/temp_0_4

INFO: Setting read mode to /tpccdata/icust2_0_30
INFO: Setting read mode to /tpccdata/ordr_0_32
INFO: Setting read mode to /tpccdata/iordr2_0_44
INFO: Setting read mode to /tpccdata/iordr2_0_29
INFO: Setting read mode to /tpccdata/iordr2_0_47
INFO: Setting read mode to /tpccdata/icust2_0_49
INFO: Setting read mode to /tpccdata/icust1_0_2
INFO: Setting read mode to /tpccdata/istok_0_1
INFO: Setting read mode to /tpccdata/ordr_0_33
INFO: Setting read mode to /tpccdata/ordr_0_37
INFO: Setting read mode to /tpccdata/iordr2_0_37
```

Setting SmartIO caching policies for database objects

A Database Administrator (DBA) with knowledge of the database activity and usage statistics may want to adjust the SmartIO caching policies based on this information. You can set a SmartIO caching policy for a specified database object, including a named tablespace, a recent partition, or a particular datafile. You can also pin a specified database object to hold it in the SmartIO cache area.

See “Pinning and unpinning database objects” on page 56.

To set the caching policy for a specified database object

- Use the following command:

```
# sfcache app [cachearea=cachearea_name] oracle -S $ORACLE_SID \
-H $ORACLE_HOME -o set --cachemode={nocache|read} \
{--datafile=name | --partition=name \
| --tablespace=name | --tablecluster=name \
| --filelist=name}
```

Pinning and unpinning database objects

Use this procedure to pin or unpin the specified database object, including a named tablespace, table-cluster, partition or datafile.

- The `pin` operation prevents the data from being evicted from the cache. You can pin commonly used database objects so that SmartIO does not evict the data and later need to cache the data again. The pinned data is kept in the cache indefinitely, until it is deleted or explicitly unpinned.
- The `unpin` operation removes data from the pinned state. The `unpin` operation does not cause the data to be immediately evicted. SmartIO considers the data for eviction in the same way as any other data, when space is required in the cache.

To pin or unpin the specified database object

- To pin or unpin the specified database object, including a named tablespace, table-cluster, partition or datafile, use the following command:

```
# sfcache app [cachearea=cachearea_name] oracle -S $ORACLE_SID \
-H $ORACLE_HOME -o {pin | unpin} {--datafile=name \
| --partition=name | --tablespace=name}
```

Enabling and disabling caching for the database

The SmartIO plug-in for oracle allows you to enable or disable caching for the database.

You can use this operation to temporarilly disable caching for database jobs like backup or data warehouse ETL (extract, transform and load) operations. After the job completes, you can enable caching. You can enable and disable caching while the database is online.

To enable caching for the database

- Use the following command:

```
# sfcache app [cachearea=cachearea_name] oracle -S $ORACLE_SID \
            -H $ORACLE_HOME -o enable
```

For example:

```
$ sfcache app cachearea=sfcachearea_1 oracle -S $ORACLE_SID \
            -H $ORACLE_HOME -o enable
INFO: Oracle Instance tpcc is running
INFO: Store DB details at /tpccdata/.CACHE_INFO
INFO: Setting enable mode to /tpccdata

INFO: Setting enable mode to /tpcclog
```

To disable caching for the database

- Use the following command:

```
# sfcache app [cachearea=cachearea_name] oracle -S $ORACLE_SID \
            -H $ORACLE_HOME -o disable
```

For example:

```
$ sfcache app cachearea=sfcachearea_1 oracle -S $ORACLE_SID \
            -H $ORACLE_HOME -o disable
INFO: Oracle Instance tpcc is running
INFO: Store DB details at /tpccdata/.CACHE_INFO
INFO: Setting disable mode to /tpccdata

INFO: Setting disable mode to /tpcclog
```

Listing cache policy details for the database

Use this procedure to list caching policies for the specified database object, including a datafile, partition or tablespace.

To list caching policies for the database

- Use the following command:

```
# sfcache app [cachearea=cachearea_name] oracle -S $ORACLE_SID \
-H $ORACLE_HOME -o list {--datafile=name \
| --partition=name | --tablespace=name}
```

For example, to list the caching policies for the tablespace stok_0:

```
$ sfcache app cachearea=sfcachearea_1 oracle -S $ORACLE_SID \
-H $ORACLE_HOME -o list --tablespace=stok_0
INFO: Oracle Instance tpcc is running
INFO: Store DB details at /tpccdata/.CACHE_INFO
FILENAME                        MODE      PINNED    CACHE_USED
--------                        ----      ------    ----------
/tpccdata/stok_0_0              read      no         1.05 GB
/tpccdata/stok_0_2              read      no        1.046 GB
/tpccdata/stok_0_1              read      no        1.055 GB
/tpccdata/stok_0_4              read      no         1.03 GB
/tpccdata/stok_0_3              read      no        1.043 GB
/tpccdata/stok_0_5              read      no        1.055 GB
/tpccdata/stok_0_6              read      no        1.044 GB
/tpccdata/stok_0_8              read      no        1.054 GB
/tpccdata/stok_0_7              read      no        1.048 GB
/tpccdata/stok_0_9              read      no         1.03 GB
/tpccdata/stok_0_10             read      no        1.029 GB
/tpccdata/stok_0_12             read      no         1.05 GB
/tpccdata/stok_0_11             read      no        1.045 GB
```

Listing cache statistics for the database

A DBA can monitor the cache usage and hit ratio at a database level. You can use the statistics to evaluate the current cache policies. If the hit ratio is not satisfactory, you can consult Oracle AWR reports and change the policies.

To list cache statistics for a database

- To view cache statistics for all of the mountpoints of the database, use the following command:

```
$ sfcache app [cachearea=cachearea_name] oracle -S $ORACLE_SID \
-H $ORACLE_HOME -o stat

INFO: Oracle Instance tpcc is running
INFO: Store DB details at /tpccdata/.CACHE_INFO
                VxFS SmartIO Statistics
               --------- -----------------------
Cache Size = 1.096 TB
 Cache Utilization = 64.3 GB ( 5.73 %)
Mount       Hit Ratio Cached Pinned  Read      Written  Data
                      Files  Files   Bytes     Bytes    Pinned
-----       --------- -----  ------  --------  -------- ------
/tpccdata   67.80 %   445    10      134.4 GB  179.0 GB 160 KB
/tpcclog    38.10 %   9      0       4 KB      8 KB     0 KB
```

About SmartIO caching for databases on VxVM volumes

SmartIO provides application templates to optimize caching for databases running on VxVM volumes. SmartIO uses the templates to apply policies to particular types of volumes in the database. For example, index volumes may have different caching policies from data volumes.

SmartIO provides the following application templates for VxVM volumes:

- Template name: oracle
  Optimizes caching for Oracle databases running on VxVM volumes.
- Template name: sybase
  Optimizes caching for Sybase databases running on VxVM volumes.

Applying a SmartIO database caching template for a VxVM volume

SmartIO provides application templates to optimize caching for databases running on VxVM volumes. SmartIO uses the templates to apply policies to particular types of volumes in the database. For example, caching policies for index volumes and data volumes.

To apply a SmartIO sybase caching template for a VxVM volume

1. Log in as root user.
2. Export the SYBASE environment variable.

   ```
   # export SYBASE=/sybase
   ```

   Where `/sybase` is the Sybase home directory.

3. Apply a SmartIO sybase caching template for a VxVM volume using the following command:

   ```
   # sfcache app cachearea=cachearea_name sybase \
    {olap|oltp} db_username db_server [db_name]
   ```

   Where:

   `olap` or `oltp` indicates the type of application load. OLAP, or Online Analytical Processing, applications process workloads intended for multi-dimensional analytical queries. OLTP, or Online Transaction Processing, applications process transaction-oriented workloads, such as for data entry and retrieval transaction processing.

   db_user_name indicates the Database User Name.

   db_server indicates the Database Server Name.

   db_name indicates the database name. The database name is optional. If database name is not specified, then the command runs for all databases under the user *db_user_name*.

   For example:

   ```
   # sfcache app cachearea=sfcachearea_1 sybase olap sa VXVMSYBS vmdb
   ```

4. Enter the password for the database user at the prompt.
5. If required, you can view the command activity in the caching log file.

   The log file is`/etc/vx/log/sfcache.log`.

   For example:

   ```
   Enabling caching for sybdg/DB1DATA
   sfcache enable sybdg/DB1DATA
   Disabling caching for sybdg/DB1LOG
   Sfcache disable sybdg/DB1LOG
   ```

To apply a SmartIO oracle caching template for a VxVM volume

1. Log in as root user.
2. Apply a SmartIO oracle caching template for a VxVM volume using the following command:

```
# sfcache app cachearea=cachearea_name oracle \
 {olap|oltp} oracle_user_name ORACLE_HOME \
 ORACLE_SID [ASM_HOME [ASM_SID]]
```

Where:

olap or oltp indicates the type of application load. OLAP, or Online Analytical Processing, applications process workloads intended for multi-dimensional analytical queries. OLTP, or Online Transaction Processing, applications process transaction-oriented workloads, such as for data entry and retrieval transaction processing.

oracle_user_name indicates the user name of the Oracle user.

ORACLE_HOME indicates the directory where the Oracle software is installed. Usually the ORACLE_HOME value is stored in the `bash.rc` file or the `profile` file of the Oracle user.

ORACLE_SID indicates the System ID used to uniquely identify the database.

ASM_HOME indicates the directory where the ASM software is installed.

ASM_SID indicates the System ID for the ASM instance. By default, the value is `+ASM`. For multiple instances of ASM, the value may differ.

The following examples show the variations of the app command for the Oracle database.

The first example shows the Oracle database created directly on VxVM volumes. In this scenario, the ASM_HOME and ASM_SID values are not applicable.

Example of Oracle on Raw volumes:

```
# sfcache app cachearea=sfcachearea_1 oracle olap oracle /ora_base/db_home rawdb
```

The next example shows the Oracle ASM database created on VxVM volumes. In this scenario, you must specify the ASM_HOME. If required, specify the ASM_SID.

Example of Oracle ASM:

```
# sfcache app cachearea=sfcachearea_1 oracle oltp oracle /orabin/dbbase/dbhome
\ testdb /orabin/gridhome
```

3 Enter the password for the database user at the prompt.

4 If required, you can view the command activity in the caching log file.

The log file is `/etc/vx/log/sfcache.log`.

Example log file for Oracle on Raw volumes:

```
Fri Jun  7 22:04:31 IST 2013 sfcache app cachearea=sfcachearea_1
oracle olap oracle /ora_base/db_home rawdb
Enabling caching for rawdg/rawvol02
/usr/sbin/vxprint -v -g rawdg -e 'v_name="rawvol02"'
/usr/sbin/vxprint -g rawdg -F %cache_area_type rawvol02
/usr/sbin/vxprint -g rawdg -F %iscachevol rawvol02
/usr/sbin/vxprint -g rawdg -F %caching rawvol02
/usr/sbin/vxprint -o alldgs -q -v -e 'v_cachearea_vm=on'
Enabling caching for rawdg/rawvol06
/usr/sbin/vxprint -v -g rawdg -e 'v_name="rawvol06"'
/usr/sbin/vxprint -g rawdg -F %cache_area_type rawvol06
/usr/sbin/vxprint -g rawdg -F %iscachevol rawvol06
/usr/sbin/vxprint -g rawdg -F %caching rawvol06
/usr/sbin/vxprint -o alldgs -q -v -e 'v_cachearea_vm=on'
Disabling caching for rawdg/rawvol01
/usr/sbin/vxprint -v -g rawdg -e 'v_name="rawvol01"'
/usr/sbin/vxprint -g rawdg -F %cache_area_type rawvol01
/usr/sbin/vxprint -g rawdg -F %iscachevol rawvol01
/usr/sbin/vxprint -g rawdg -F %caching rawvol01
```

Example log file for Oracle ASM:

```
Enabling caching for testdg/testvol
/usr/sbin/vxprint -v -g testdg -e 'v_name="testvol"'
/usr/sbin/vxprint -g testdg -F %cache_area_type testvol
/usr/sbin/vxprint -g testdg -F %iscachevol testvol
/usr/sbin/vxprint -g testdg -F %caching testvol
/usr/sbin/vxprint -o alldgs -q -v -e 'v_cachearea_vm=on'
Enabling caching for testdg/testvol2
/usr/sbin/vxprint -v -g testdg -e 'v_name="testvol2"'
/usr/sbin/vxprint -g testdg -F %cache_area_type testvol2
/usr/sbin/vxprint -g testdg -F %iscachevol testvol2
/usr/sbin/vxprint -g testdg -F %caching testvol2
/usr/sbin/vxprint -o alldgs -q -v -e 'v_cachearea_vm=on'
```

Technology Preview: Distributed SmartIO in Veritas InfoScale storage environments

Distributed SmartIO is a new feature available as a technology preview in Veritas InfoScale for configuration and testing in non-production environments. It is primarily targeted for Oracle RAC or ODM.

Note: All the configurations which are supported on SmartIO read caching for applications running on VxFS file systems are also supported with Distributed SmartIO.

With the advancements in the hardware technology - network interconnect such as infiniband, accessing and sharing the data using the network rather than the disk as a medium for data sharing is proving to be faster and cost efficient in storage environments. Data can be cached on faster but costlier SSD storage on few nodes in a cluster. High speed network interconnect can be used to fetch the data as required on any node in the cluster.

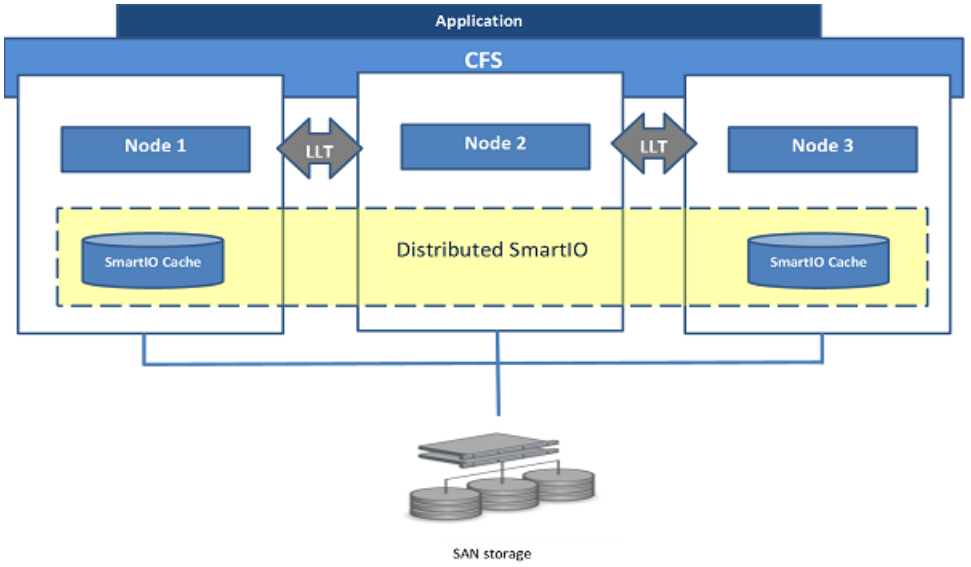
Considering these benefits, Veritas InfoScale has come up with a robust solution, **Distributed SmartIO**, which lets you share SSD resources between all the nodes in the cluster for caching frequently read data.

Distributed SmartIO feature uses SSD resources both local and remote for caching purposes to improve performance of the applications running within the cluster. Using Distributed SmartIO, you can create a coherent cache area using SSD across all the nodes in the cluster irrespective of whether each node has an independent local cache.

In the Distributed SmartIO feature, a few or all the nodes in the cluster have SmartIO cache attached to it. Each node within the cluster is connected through a high speed network interconnect such as Infiniband or 10G Ethernet. Read Operation (RO) cached data and metadata about the cached data is distributed and shared across the SmartIO cache within all nodes in the cluster. The metadata also resides at the back-end storage device.

See Read Operations (RO) in Distributed SmartIO

See Write Operations in Distributed SmartIO

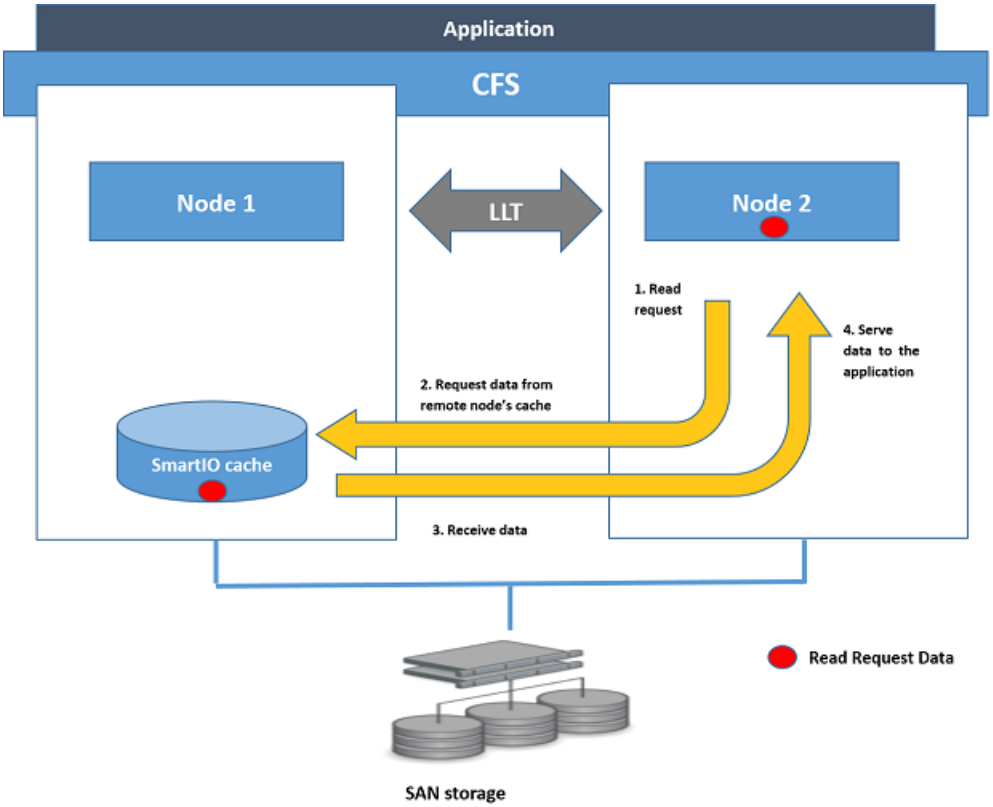


Read Operations (RO) in Distributed SmartIO

In a clustered environment, when the application issues a RO using Direct I/O, Distributed SmartIO checks to see if the RO can be serviced from the local cache on the existing node. In the absence of requested data from local cache on the existing node, Distributed SmartIO attempts to serve the RO from remote node's local cache. If the requested data is not present in any cache, then the data is rendered asynchronously in the Distributed SmartIO from the back-end storage.

Note: You can create a coherent cache area using SSDs on all nodes in the cluster even if one or more nodes within the cluster does not have an independent SmartIO (Local Cache) attached to it.

The following diagram gives an overview of how the read operations are processed in Distributed SmartIO:

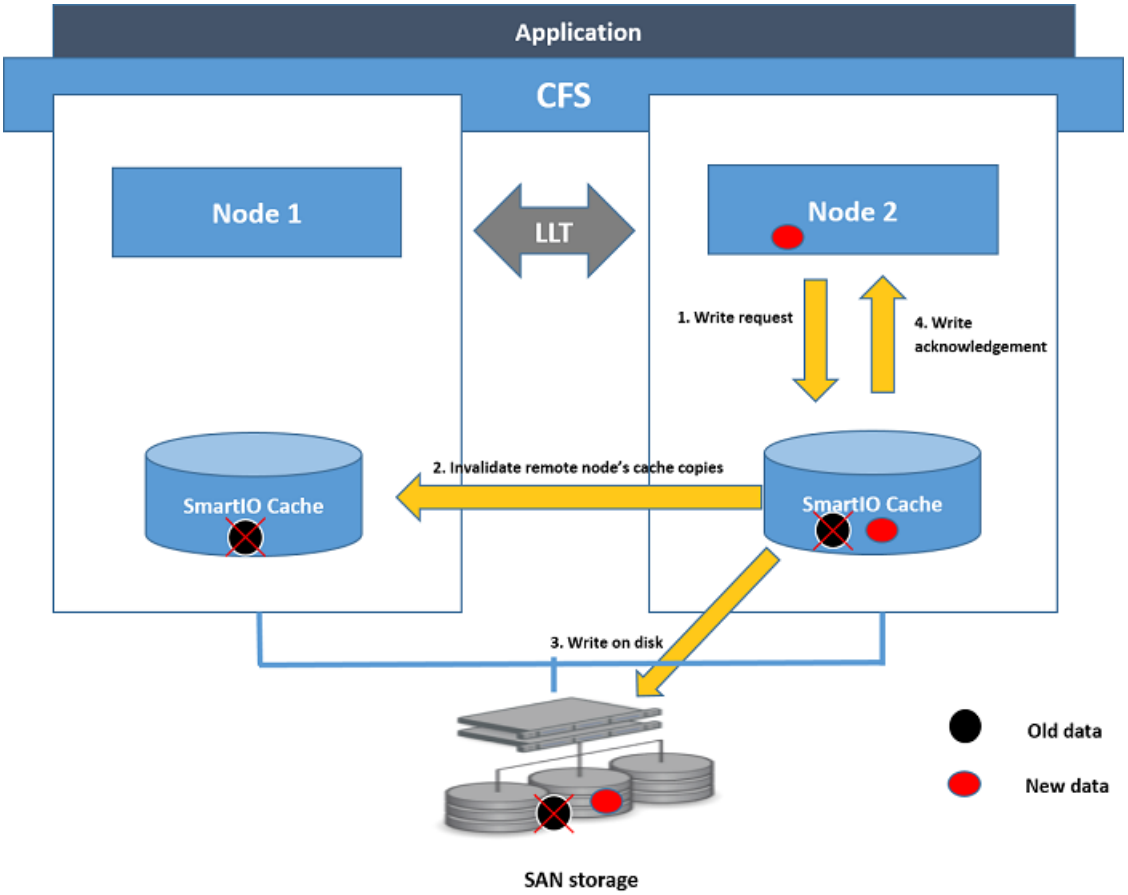


Write Operations in Distributed SmartIO

In a clustered environment, when the application issues a write operation on any node within the cluster, it invalidates copies of the data from all the remote caches except the data on local cache of the node on which the write operation is performed. The local cache of this node is updated with the new data.

Note: You can create a coherent cache area using SSDs on all nodes in the cluster even if one or more nodes within the cluster does not have an independent SmartIO (Local Cache) attached to it.

The following diagram gives an overview of how the write operations are processed in Distributed SmartIO:


Application
CFS
Node 1
LLT
Node 2
1. Write request
4. Write acknowledgement
SmartIO Cache
2. Invalidate remote node's cache copies
SmartIO Cache
3. Write on disk
Old data
New data
SAN storage

Enabling Distributed SmartIO feature in Veritas InfoScale storage environment

To enable Distributed SmartIO feature in Veritas InfoScale storage environment

1. Use `sfcache` command to create SmartIO cache on the required nodes in the cluster.

   See "Creating a cache area " on page 69. for more details.

   See `sfcache(1M)` manual page for more details.

2. While mounting a VxFS file system, use `cluster,smartiomode=cfusion` option to enable Distributed SmartIO.

   ```
   # mount -t vxfs /dev/vx/dsk/testdg/vol1 -o cluster,
   smartiomode=cfusion /mnt1
   ```

3. To see statistics on the cache usage, use the following command:

   ```
   # sfcache stat
   ```

```
Cache Name: cache_dg1/cachevol
Cache Size:   24.97 GB
Cache Utilization:    70.8 MB ( 0.28 %)
File Systems Using Cache:       6
Writeback Cache Use Limit:  Unlimited
Writeback Flush Timelag:     10 s

Read Cache                              Writeback
Hit Ratio   Data Read  Data Written   Hit Ratio  Data Written  rdcachename   wbcachename

Total:
45.40 %     4.99 MB     2 MB           0.00 %     0 KB

/mnt1:
74.98 %     2.996 MB    1 MB           0.00 %     0 KB        cache_dg1/cachevol     -
Remote Read:
66.62 %     1.996 MB
```

The output displays the **Data Read** and **Hit Ratio** statistics under the **Remote Read** section for a particular mount point. It provides statistical information about the data that is served by SmartIO cache (Local Cache) for reads happening on the remote nodes.

Limitations of Distributed SmartIO

Support for Distributed SmartIO is limited by the following constraints:

- Distributed SmartIO feature is supported only for applications running on VxFS file system
- Distributed SmartIO feature is beneficial only when used with Direct I/O

Chapter 3

Administering SmartIO

This chapter includes the following topics:

- Creating a cache area
- Displaying information about a cache area
- Enabling or disabling caching for a data object
- Adding a device to the cache area
- Pausing caching from a volume to a cache area
- Removing a device from the cache area
- Destroying a cache area
- Setting the attributes of the VxVM cache area
- Setting or changing the caching mode for a VxFS cache area
- Flushing dirty data from a writeback cache area
- Tuning the writeback caching
- Viewing the SmartIO cache statistics

Creating a cache area

SmartIO introduces the concept of a cache area. The cache area is the storage space that SmartIO uses to store the cached data and the metadata about the cached data. You create a cache area to use for I/O caching. Usually, you use SSD devices or other fast devices for the cache area. A cache area can be used for VxFS caching or VxVM caching. You can create multiple cache areas for VxFS caching on each system and one cache area for VxVM caching on each system.

To create a cache area on a device, specify the device name (disk access name) or the name of a disk group and volume on the device.

By default, a cache area has an association type of auto. All of the data objects of the same type (VxVM or VxFS) are implicitly associated with the auto cache area. If the association type is noauto, you must explicitly associate the data objects to the cache area.

In a cluster, each node has a separate, local cache area.

To create a VxVM cache area

- Create the VxVM cache area on the SSD device, using one of the following commands:
  - Specify one or more devices using the disk access name (*daname*) of the device. The device should be initialized for VxVM use and have the `cdsdisk` format.

    ```
    # sfcache create -t VxVM [size] daname[...] \
    [cacheline_size=cacheline_size] [--auto|--noauto]
    [--nostripe|ncols=N] \[cachearea_name]
    ```

    Where:
    daname specifies the disk access name of the device on which to create the cache area.
    cachearea_name specifies a customized name for the cache area. If unspecified, the SmartIO feature automatically generates a name for the cache area.
    size specifies the size for the cache area. By default, the cache area uses all of the available space on the device.
    `--noauto|--auto` specifies the type of cache area. The default is `--auto`.
    `--nostripe|ncols=`*n* specifies the layout options for the cache area. By default, the cache area is created over a striped volume if two or more disks are specified. Use the `ncols=`*n* option to specify the number of columns for the striped volume. Use the `--nostripe` option to create the cache area on a concatenated volume over the specified disks.
    cacheline_size specifies the unit that SmartIO uses for caching. When the application I/O accesses the data, the SmartIO moves the data to the cache according to the cacheline size. Generally, you do not need to change the *cacheline_size*.
    For example:

    ```
    # sfcache create -t VxVM ssd0_0
    ```

If you specify more than one disk, the cache area is striped across the specified disks by default. For example:

```
# sfcache create -t VxVM ssd0_0 ssd0_1
```

- Alternatively, you can specify the name of a disk group and a volume to use for the cache area. In this case, you cannot specify a size: the cache area uses the entire volume.

  ```
  # sfcache create -t VxVM  [cacheline_size=cacheline_size] \
   [--noauto|--auto] dg/vol
  ```

  Where:
  dg/*vol* specifies the disk group name and volume name that you want to use for the cache area. SmartIO uses this name for the cache area.
  `--noauto|--auto` specifies the association type of the cache area. The default is `--auto`.
  For example:

  ```
  # sfcache create -t VxVM --auto ssd_dg/ssd_vol
  ```

To create a VxFS cache area

- Create the VxFS cache area on the SSD device, using one of the following commands.
  - Specify one or more devices using the disk access name (*daname*) of the device. The device should be initialized for VxVM use and have the `cdsdisk` format.

    ```
    # sfcache create [-t VxFS] [size] daname[...] [--auto|--noauto] \
     [--nostripe|ncols=N] [cachearea_name]
    ```

    Where:
    daname specifies the disk access name of the device on which to create the cache area.
    cachearea_name specifies a customized name for the cache area. If unspecified, the SmartIO feature automatically generates a name for the cache area.
    size specifies the size for the cache area. By default, the cache area uses all of the available space on the device.
    `--noauto|--auto` specifies the type of cache area. The default is `--auto`.
    For example:

    ```
    # sfcache create ssd0_0
    ```

`--nostripe|ncols=`*n* specifies the layout options for the cache area. By default, the cache area is created over a striped volume if two or more disks are specified. Use the `ncols=`*n* option to specify the number of columns for the striped volume. Use the `--nostripe` option to create the cache area on a concatenated volume over the specified disks.

- Alternatively, you can specify the name of a disk group and a volume to use for the cache area. In this case, you cannot specify a size: the cache area uses the entire volume.

  ```
  # sfcache create [-t VxFS] [--noauto|--auto] dg/vol
  ```

  Where:
  dg/*vol* specifies the disk group name and volume name that you want to use for the cache area. SmartIO uses this name for the cache area.
  `--noauto|--auto` specifies the type of cache area. The default is `--auto`.
  For example:

  ```
  # sfcache create --auto ssd_dg/ssd_vol
  ```

To create an FEL cache area

- Create an FEL cache area on an SSD device, using one of the following commands.

  Specify one or more devices using the disk access name (daname) of the device. The device should be initialized for VxVM use and should have the cdsdisk format.

  - `# sfcache create [`*size*`] daname[...] [ --fel [dgname=name] [nmirrors=N] [node1:`*daname node2*`:`*daname*`]] [`*cachearea_name*`]`
    Where:
    daname specifies the disk access name of a device on which you want to create a cache area.
    cachearea_name specifies a customized name for an FEL cache area. If unspecified, SmartIO automatically generates a name.
    size specifies the size for a cache area. By default, the cache area uses all of the available space on the device.
    dgname specifies the name of an already created FSS disk group.
    nmirrors specifies the number of mirrored volumes to be created on FSS diskgroup.
    node1,*node2* specifies the cluster node names.
    For example:

    ```
    # sfcache create --fel ssd0_0
    ```

    In SFCFSHA configuration:

```
# sfcache create --fel node1:ssd0_0 node2:ssd1_0
```

- Alternatively, specify the name of an FSS disk group and a volume to use for an FEL cache area. In this case, you cannot specify a size. The FEL cache area uses the entire volume.

  For example:

  ```
  # sfcache create [--fel] dg/vol
  ```

  Where:

  dg/vol is the FSS disk group name and volume name. SmartIO uses this name for the FEL cache area.

  For example:

  ```
  # sfcache create --fel ssd_dg/ssd_vol
  ```

Displaying information about a cache area

SmartIO uses a cache area on the SSD devices for caching. Each cache area has attributes such as the cache type, size, association type, the caching state, and the devices used for caching. The cache area also has associated data objects.

For a VxFS cache area, the `sfcache list` command shows the caching mode for the file or directory. If the mode is not explicitly set, the file or directory inherits the caching mode of the mount point. If the mode is explicitly set for a file or directory, that value persists across remounts. The displayed caching mode may differ from the mode that is enabled for the mount point. The `writeback` mode is not enabled unless the file system is mounted in `writeback` mode. If a file or directory is set to `writeback` mode, but the file system is mounted in another mode, the file or directory inherits the caching mode of the mount point.

To display information about a cache area

1 Use the following command to display information about the cache areas on the system.

```
# sfcache list

NAME          TYPE SIZE   ASSOC-TYPE STATE  FSTYPE  DEVICE
sfcachearea_1 VxVM 9.91g  AUTO       ONLINE   -     ssd0_2809
sfcachearea_2 VxFS 31.97g AUTO       ONLINE reserve ssd0_0
```

2 Use the following command to display information about a specific cache area.

Example of a VxVM cache area:

```
# sfcache list sfcachearea_1
Cachearea: sfcachearea_1
Assoc Type: AUTO
Type: VxVM
Size: 30.00g
Cacheline Size: 64.00k
Memory Size: 16.00m
State: ONLINE
Layout: CONCAT
Number of Columns: 0

ASSOCIATED DATA OBJECTS:

ASSOC DATAOBJECT NAME   CACHING-MODE  STATE    KSTATE
testdg/testvol1         read          ENABLED  ENABLED
testdg/testvol2         read          ENABLED  ENABLED
```

Example of a VxFS cache area:

```
# sfcache list sfcachearea_2
Cachearea: sfcachearea_2
Assoc Type: AUTO
Type: VxFS
Size: 7.00g
State: ONLINE
Layout: CONCAT
FStype: default
Number of Columns: 0

FSUUID                      SIZE      MODE       MOUNTPOINT     CACHENAME
a6178a5604a50200577a...    759.0 MB  read       /ora_inst1     sfcachearea_2
0d929856a7d60e003d38...    512.0 MB  writeback  /pdb2          sfcachearea_2
90929856b68800008141...     50.1 MB  read       /fast_recovery sfcachearea_2
02188a569d7e0700567d...        4 KB  nocache    -              sfcachearea_2
31188a5679a90900987d...        4 KB  nocache    -              sfcachearea_2
a55b8a56db160a00fa61...        4 KB  nocache    -              sfcachearea_2
```

Note: To display information about an FEL cache area:

See "Verifying the VxFS cache area and monitoring FEL-based write-back caching " on page 40.

Enabling or disabling caching for a data object

With the SmartIO feature, you can disable or enable caching for a volume or a mount point.

If automatic caching is configured, you do not need to explicitly enable caching for any data objects. The SmartIO feature caches all of the data objects of the same type as the cache area (VxVM or VxFS). The data objects have the caching state of `auto`, by default. If required, you can disable caching for a particular data object. If the system has an `auto` cache area, the SmartIO feature caches any objects that have the `auto` state or the `enabled` state.

If automatic caching is not configured, you must explicitly enable caching for any data objects that you want the SmartIO feature to cache. If the system has a no-auto cache area, the SmartIO feature only caches the objects that have the `enabled` state. The SmartIO feature does not cache data objects with the caching state of `auto`, which is the default caching state.

See "Enabling or disabling caching for a file system" on page 76.

See "Enabling or disabling caching for a data volume " on page 77.

Enabling or disabling caching for a file system

The SmartIO feature enables you to disable or enable caching for a specific file system. For a cluster file system, you must run the commands on each node in the cluster.

Enabling or disabling caching for a file system is not persistent when the cache area is brought offline and online, or when you shrink a cache area. In this case, the caching for the file system depends on the caching mode set on the mount point or set with the `sfcache set mode` command. To make the caching status persistent, use the `sfcache set mode` command.

See "Setting or changing the caching mode for a VxFS cache area" on page 81.

To enable caching for a file system

- To enable caching for a file system, use the following command.

  ```
  # sfcache enable mount_point
  ```

  Where:

  mount_point is the mount point for the file system.

To disable caching for a file system

- To disable caching for a file system, use the following command. Use the `-o purge` option to clear the existing cache data from the cache. By default, the data is not purged.

  ```
  # sfcache disable [-o purge] mount_point
  ```

  Where:

  mount_point is the mount point for the file system.

To purge the caching data for a file system

- To purge the caching data for a file system, use the following command. Use the `purge` option to clear the existing cache data from the cache.

  ```
  # sfcache purge {mount_point|fsuuid}
  ```

  Where:

  mount_point is the mount point for the file system.

  fsuuid specifies the UUID of the file system. Specify the *fsuuid* to purge the caching data for a file system that is not mounted.

Enabling or disabling caching for a data volume

SmartIO enables you to disable or enable caching for a specific data volume.

To enable caching for a specific data volume

- To enable caching for a specific data volume, use the following command.

  ```
  # sfcache enable [--auto] dgname/volname
  ```

To disable caching for a specific data volume

- To disable caching for a specific data volume, use the following command.

  ```
  # sfcache disable dgname/volname
  ```

Adding a device to the cache area

You can add a device to the cache area when you increase the size of the cache area. This operation can be done online.

To resize a cache area

- To resize a cache area, use the following command. Specify a new device by the disk access name to expand the cache area onto that device. The `maxsize` option increases the cache area to the maximum size supported by the devices in the cache area.

  ```
  # sfcache resize [daname ...] {newsize | maxsize} cachearea_name
  ```

Pausing caching from a volume to a cache area

For VxVM type of cache areas, you can temporarily stop caching to a cache area for a particular volume, without removing the association between the volume and the cache. You can pause the caching and resume it later. When caching is paused, no new data is cached on reads. The contents of the cache are updated if writes occur to a region that is already cached.

To pause caching

- To pause, use the following command.

  ```
  # sfcache set dgname/volname --pause
  ```

To resume caching

- To resume caching for the volume, use the following command.

  ```
  # sfcache set dgname/volname --resume
  ```

Removing a device from the cache area

If the cache area spans more than one device, you can remove one of the devices from the cache area.

To remove a device from a cache area

1. Make sure that the cache will fit on the remaining devices after you remove the device. If necessary, resize the cache to a size smaller than the size of the remaining devices.

   If you shrink a VxFS cache area, all of the cached data (including statistics) might be evicted.

   ```
   # sfcache resize newsize cachearea_name
   ```

   For example, suppose the current cache area `sfcachearea_1` has a maximum size of 5G. You want to remove an SSD `ssd0_0` that is 2G. First, resize the cache area to 2.5G:

   ```
   # sfcache resize 2.5g sfcachearea_1
   ```

2. To remove a device from a cache area, specify the device by the disk access name to remove the cache area.

   ```
   # sfcache rmdev [daname ...]
   ```

   For example, to remove the SSD `ssd0_0`:

   ```
   # sfcache rmdev ssd0_0
   ```

3. Optionally, you can use the `maxsize` option to resize the cache area to the maximum size that fits on the remaining devices in the cache area.

   ```
   # sfcache resize maxsize cachearea_name
   ```

   For example:

   ```
   # sfcache resize maxsize sfcachearea_1
   ```

Destroying a cache area

You can destroy a cache area, which removes all data from the cache. Before you destroy the cache area, you must bring the cache area offline so that no caching is taking place.

To destroy the cache area

1 To bring the cache area offline:

```
# sfcache offline cachearea_name
```

2 To destroy the cache area, use the following command.

```
# sfcache delete cachearea_name
```

Setting the attributes of the VxVM cache area

For a VxVM cache area, you can set the following attributes:

- `memsz`. The amount of system RAM that the SmartIO solution requires to keep VxVM caching-related metadata. When the VxVM cache area is created, SmartIO calculates the best possible value. You do not need to change this value, unless the system is low on system RAM.
- association type. The association type is either `--auto` or `--noauto`. The association type determines whether caching is automatically enabled for volumes on the system.
  See "Automatic caching for VxVM volumes" on page 14.

To set the memory size

- To set the memory size, use the following command.

  ```
  # sfcache set {dg/vol|cachearea_name} memsz=size
  ```

  Where

  dg/*vol* specifies the disk group and volume that is used for the cache area.

  cachearea_name specifies the name of the cache area

  size specifies the maximum size for the memory for in-core metadata.

  For example:

  ```
  # sfcache set myspecialcache memsz=128m
  ```

To set the association type

- To set the association type, use the following command.

  ```
  # sfcache set {--auto|--noauto} {dg/vol|cachearea_name}
  ```

  Where

  dg/*vol* specifies the disk group and volume that is used for the cache area.

  cachearea_name specifies the name of the cache area

  For example:

  ```
  # sfcache set --noauto mydg/myvol
  ```

Setting or changing the caching mode for a VxFS cache area

For a VxFS cache area, the caching mode determines what kind of caching is performed for the specified mount point. The mode can be `nocache`, `read`, or `writeback`. The default mode is `read`.

A VxVM cache area only supports read mode.

You cannot change the type (VxVM or VxFS) of a cache area. You must destroy the cache area and create a new cache area of the required type.

You can set the caching mode of a VxFS mount point with the `-o smartiomode` option. The caching mode that is set with the mount command represents the highest level of caching that can be enabled for objects on the mount point. If you specify nocache mode, the SmartIO caching is disabled for the mount point. You cannot enable SmartIO caching for any data objects in that mount point. You must remount the file system to enable caching.

Similarly, if you specify read mode during the mount, you cannot enable SmartIO `writeback` caching for any data objects in that mount point.

To set the caching mode of a VxFS mount point

- To set the caching mode when you mount the VxFS file system, use the following command:

  ```
  # mount -t vxfs -o smartiomode=[mode] /dev/vx/dsk/testdg/vol1 /mnt1
  ```

  Where:

  mode is one of the following:

- writeback
- read
- nocache

To change the caching mode of a VxFS mount point

- To change the caching mode of a file or directory, use the following command. You cannot change the caching mode to a higher level of caching than the mode set on the *mount_point*. For example, you cannot enable read caching if you specified the mode as `nocache` when you mounted the file system.

  ```
  # sfcache set [-r] mode=[nocache|read|writeback] {file|dir}
  ```

  Use the `-r` option to make the change recursive.

Flushing dirty data from a writeback cache area

With SmartIO, dirty data in the cache is automatically flushed to the disk during normal operations. The dirty data is flushed when the file system is unmounted, or during other operations that require a file system freeze. The dirty data is also flushed periodically at intervals. You can control the interval by configuring the tunable parameters.

See "Tuning the writeback caching" on page 82.

Disabling `writeback` caching for a file also flushes any write-back dirty data for that file.

In some cases, you may want to manually trigger flushing of the dirty data from the cache to the disk. For example, to ensure data consistency, you would flush the cache before you create an array level snapshot.

You can manually trigger flushing of the dirty data using the following command.

```
# sfcache flush [-r] {mount_point|directory|file}
```

Use the `-r` option to make the selection recursive.

Tuning the writeback caching

When `writeback` caching is enabled, any data that is read from the disk is cached, unless the file is explicitly marked for "no caching" or if the cache is full. For writes, certain writes cause the data to be cached. You can load a file to speed up the application. Pinning a file in the cache ensures that the data does not get evicted. If some data is already cached, and that portion of the disk is overwritten, then

SmartIO also writes the new data to the cache device to ensure that the cached data remains up to date.

If you are using a database template, SmartIO caches according to the template rules.

See “About SmartIO caching for Oracle databases on VxFS file systems ” on page 52.

You can use the following tunable parameters to adjust the size of the cache and how long data is held in the cache.

Setting the maximum space used for dirty data per node

Setting the maximum retention time used for dirty data

Setting the maximum space used for dirty data per node

When `writeback` is enabled, you can configure how much of the cache is used for dirty data. The `writeback_size` attribute sets the maximum amount of cache area space that is used for writeback data for each file system. The maximum is set per node. By default, there is no maximum. If you configure a maximum, the value must be at least 512 MB.

For a cluster file system, SmartIO in `writeback` mode reflects, or mirrors, the cache data for each node to the other node’s SSD cache. The actual disk space usage is twice the amount used for a standalone file system. The reflected data is not considered in the maximum size, however. For example, if the `writeback_size` is set to 512 MB, a cluster file system uses up to 512 MB on each node, for a total of 1024 MB.

Run the following command to configure the maximum. For a cluster file system, run the command on each node of the cluster to make the setting cluster wide.

```
# sfcache set writeback_size=size
```

For example:

```
# sfcache set writeback_size=1g
```

Use the following command to view the current value:

```
# sfcache stat cachearea_name
```

See “Viewing the SmartIO cache statistics ” on page 84.

Setting the maximum retention time used for dirty data

Dirty data is data in the cache that has not been flushed to the disk and so is out of sync with the data disk. The retention time determines how long the dirty data might remain unflushed. The default is 10 seconds.

For a cluster file system, run the command on each node of the cluster to make the setting cluster wide.

```
# sfcache set writeback_interval=interval
```

For example:

```
# sfcache set writeback_interval=100
```

Use the following command to view the current value:

```
# sfcache stat cachearea_name
```

See “Viewing the SmartIO cache statistics ” on page 84.

Viewing the SmartIO cache statistics

Use the `sfcache stat` command to display the cache statistics for the system.

Table 3-1 describes the cache statistics in the output.

Table 3-1 Cache statistics

Field	Description
HIT RATIO (VxVM cache)	Percentage of total I/Os that are satisfied from the cache. Displayed for reads and writes.
ART(Hit)ms (VxVM cache)	Average response time for I/Os that are satisfied from the cache.Displayed for reads (RD) and writes (WR).
ART(Miss)ms (VxVM cache)	Average response time for I/Os that are not satisfied from the cache. Displayed for r reads (RD) and writes (WR) .
BYTES (VxVM cache)	Total size of I/Os for reads (RD) and writes (WR).
NAME (VxVM cache)	Name of the cache area.
TYPE (VxVM cache)	Whether the cache area is VxVM or VxFS.

Table 3-1 Cache statistics *(continued)*

Field	Description
%CACHE (VxVM cache)	Percentage of the cache area that is currently used for data.for all data objects.
Cache Size (VxFS cache)	Size of the cache area.
Cache Utilization (VxFS cache)	Percentage of the cache area that is currently used for data.
File Systems Using Cache (VxFS cache)	Number of file systems using the cache.
Writeback Cache Use Limit (VxFS cache)	Size of the cache area that is used for writeback. You can set the size with the `writeback_size` attribute. If no size is set, the field displays Unlimited. See "Tuning the writeback caching" on page 82.
Writeback Flush Timelag	Interval between when the data is written to the cache and when it is flushed to the disk. If the Writeback Flush Timelag is small, such as 10 seconds, then sfcache statistics will not show. Data is flushed to disk faster. In this case, you can determine the caching usage based on the WB Hit Ratio.
Hit Ratio (VxFS cache)	Percentage of total I/Os that are satisfied from the cache. Displayed for reads and writes.
Data Read (VxFS cache)	Data read from the cache.
Data Written (VxFS cache)	Data written to the cache.
Files Cached (VxFS cache)	Number of files present in the cache.
Files Pinned (VxFS cache)	Number of pinned files in the cache.
Data Pinned (VxFS cache)	Amount of data pinned in the cache.

To view the cache statistics

- Use the following command:

```
# sfcache stat
```

```
                     HIT RATIO   ART(Hit)ms  ART(Miss)ms   BYTES
NAME             %CACHE RD    WR    RD    WR    RD    WR    RD     WR
TYPE: VxVM
sfcachearea_1    13.43  91.24 94.20 0.142 0.819 0.414 0.798 15.31g 4.21g

TYPE: VxFS
NAME: sfcachearea_2
               Cache Size:    48.0 GB
        Cache Utilization:    72.2 MB ( 0.15 %)
 File Systems Using Cache:       1
Writeback Cache Use Limit:  Unlimited
  Writeback Flush Timelag:      10 s

Read Cache
Hit Ratio  Data Read  Data Written  Files Cached  Files Pinned  Data Pinned
   0.00 %       0 KB          0 KB             0             0         0 KB
```

Viewing the detailed caching stats for a VxVM cache area

To view the detailed caching statistics for a VxVM cache area

- Use the following command:

```
# sfcache stat sfcachearea_1
```

```
                            HIT RATIO    ART(Hit)ms   ART(Miss)ms   BYTES
NAME                     %CACHE RD    WR     RD    WR     RD     WR     RD     WR
TYPE: VxVM
sfcachearea_1            13.43  91.24 94.20  0.142 0.819  0.414  0.798  15.31g 4.21g

ASSOCIATED DATA OBJECTS:
testdg/testvol1          6.10   90.00 96.00  0.141 0.459  0.348  0.448  6.77g  1.89g
testdg/testvol2          7.32   91.00 92.00  0.143 1.179  0.480  1.149  8.54g  2.31g
```

Viewing the detailed caching stats for a VxFS cache area

For a VxFS cache area, the statistics do not change after you unmount and mount the file systems. For a cluster file system, the statistics do not change after you reboot the cluster nodes.

To view the detailed caching statistics for a VxFS cache area

1 Use the following command:

```
# sfcache stat sfcachearea_1
```

```
TYPE: VxFS
NAME: sfcachearea_1
       Cache Name: sfcachearea_1
               Cache Size:       5 GB
        Cache Utilization:   1.426 GB (28.51 %)
File Systems Using Cache:       2
Writeback Cache Use Limit:  Unlimited
  Writeback Flush Timelag:      10 s

Read Cache                                  Writeback
Hit Ratio  Data Read Data Written Hit Ratio  Data Written  rdcachename  wbcachename

Total:
   7.98 %  157.7 MB  1.545 GB     0.00 %     0 KB

/pdb1:
   7.98 %  157.7 MB  1.545 GB     0.00 %     0 KB         sfcachearea_1       -

/pdb2:
   0.00 %  0 KB      0 KB         0.00 %     0 KB         sfcachearea_1  sfcachearea_2
```

2 To see details, use the `-l` option:

```
# sfcache stat -l
TYPE: VxFS :
       Cache Name: sfcachearea_2
               Cache Size:       7 GB
        Cache Utilization:    1.26 GB (18.00 %)
File Systems Using Cache:       6
Writeback Cache Use Limit:  Unlimited
  Writeback Flush Timelag:      10 s

       Cache Name: sfcachearea_1
               Cache Size:       5 GB
        Cache Utilization:   1.447 GB (28.94 %)
File Systems Using Cache:       2
Writeback Cache Use Limit:  Unlimited
  Writeback Flush Timelag:      10 s

.
.
.
```

Chapter 4

Troubleshooting and error handling

This chapter includes the following topics:

- Support for a persistent or 'warm' VxVM cache
- Cache area is lost after a disk failure (3158482)
- Cache is not online after a reboot
- Recovering the write-back cache after a node failure

Support for a persistent or 'warm' VxVM cache

A warm cache means that the contents of the cache remain persistent across planned reboots. By default, SmartIO does not provide a warm cache capability for VxVM caches. The cache area metadata is not flushed during a system shut down. The cache is invalidated after the system reboot, and whenever the volumes need to be restarted. This behavior is known as a 'cold' cache.

Veritas does not recommend that you configure a warm cache, because it may lead to data inconsistency. However, in some circumstances, when proper care is taken, it can be beneficial to configure a warm cache. For example, in the case of a controlled, planned reboot you may want to explicitly enable a warm cache. This feature enables you to flush the metadata to the cache, thus creating a warm cache.

If you enable a persistent or warm cache, VxVM detects and invalidates the persistent cache if the data volume was updated while the cache was offline. In clustered environments, if the cache area on a node is persisted during a planned shutdown on that node, and the data volume is updated on another node, the warm cache contents are treated as stale and persisted cached data is evicted.

To enable a warm cache for a planned reboot

1. Before the system reboot, shut down all the applications.
2. Create the warm cache by running the following command:

   ```
   # sfcache offline --flushmeta cachearea_name
   ```

   If desired, you can add this command to your shutdown script.

   You must run the above command before the application restart or system reboot. If not, the data in the cache is purged and the cache is not populated after the reboot.

Primary volume failure with a stale cache could cause possible data corruption

If the data in the primary volume needs to be restored, the existence of a warm cache may lead to the use of a stale cache. In some cases, the stale cache could cause data corruption. This situation is not common but the possibility exists if a restoration method such as the following is used:

- Restore from an array snapshot
- Restore of the primary LUN from backup using NetBackup or other backup software.
- VxVM configuration backup and restore.

Migrating a cache during HA failover is not supported

Even if a warm cache is enabled, SmartIO does not provide a capability to migrate the cache in a high availability cluster during a failover. The warm cache is not available to the application after failover.

Cache area is lost after a disk failure (3158482)

SmartIO supports multiple VxFS cache areas and one VxVM cache area. If you create one cache area, and the disk fails, the cache area becomes disabled. If you attempt to create a second cache area of the other type before the cache disk group is enabled, then the first cache area is lost. It cannot be brought online.

For example, first you created a VxFS cache area. The disk failed and the cache area is disabled. Now create the VxVM cache area. While creating VxVM cache area, SmartIO looks for an existing default cache area. Due to the failed disk, the existing cache area cannot be found. So SmartIO creates a VxVM cache area with the same name. Now even if disk containing VxFS cache area comes up, SmartIO

cannot access the original cache area. In this scenario, the VxFS cache area is lost. Losing the cache area in this case does not result into any data loss or data inconsistency issues.

Workaround:

Create a new VxFS cache area.

Cache is not online after a reboot

Generally, the SmartIO cache is automatically brought online after a reboot of the system.

If the SSD driver module is not loaded automatically after the reboot, you need to load the driver and bring the cache disk group online manually.

To bring a cache online after a reboot

1. Load the SSD driver module with the `insmod` command.

   See the Linux documentation for details.

2. Perform a scan of the OS devices:

   ```
   # vxdisk scandisks
   ```

3. Bring the cache online manually:

   ```
   # vxdg import cachedg
   ```

Recovering the write-back cache after a node failure

In `writeback` mode, SmartIO stores data in the cache before writing the data to the disk. If the device that contains the cached data fails, the unflushed data in the cache area must be flushed to the disk when the system comes back online. Until the cached data is flushed, the file data that is cached in `writeback` mode may not be completely present on the disk.

In some cases, the SmartIO cache containing dirty write-back data is not available for flushing. For example, an error may have occurred on the SSD device that contains the cache area. When the cache device comes back online, then dirty data for such files is skipped from flushing. The files that have pending dirty data are not accessible. Any I/O on those files (except deletion) returns an I/O error (EIO).

The following error message is displayed in the syslog:

`Writeback cache recovery is failed for` *`mounted_device`* `with error` *`error_code`*`:`

If you want to restore access to such files, use the following procedure.

Restoring access to files that are not accessible

1. To restore access to the files, use the following command. This command requires root privileges.

   ```
   # sfcache restore-access -r {mount_point|directory|file}
   ```

   For example:

   ```
   # /usr/sbin/sfcache restore-access /testFS
   ```

   For cluster file systems, run the `sfcache restore-access` command on each node of the cluster.

2. After you restore access to the files or directories, restore access for the mount point.

   ```
   # sfcache restore-access -r mount_point
   ```

   For example:

   ```
   # /usr/sbin/sfcache restore-access /testFS
   ```

3. To enable the write-back caching for the file system, use the following commands:

   ```
   #  sfcache disable /testFS
   # sfcache enable /testFS
   ```

Appendix A

Command reference

This appendix includes the following topics:

- SmartIO command reference

SmartIO command reference

Table A-1 lists commands for using the SmartIO feature.

See the `sfcache`(1M) manual page.

Table A-1 SmartIO command reference

Command	Description
`sfcache app`	Applies the specified template name.
`sfcache create`	Creates a cache area.
`sfcache delete`	Deletes the specified cache area.
`sfcache disable`	Disables caching for the specified data object.
`sfcache enable`	Enables caching for the specified data object.
`sfcache flush`	Flushes any write-back data for this file system or cache.
`sfcache list`	Displays the cached file systems or volumes and their cache usage.
`sfcache load`	Loads the specified file into the cache area.

Table A-1 SmartIO command reference *(continued)*

Command	Description
`sfcache maxsize`	Displays the amount of free space in the devices that are already provisioned for caching.
`sfcache offline`	Stops VxFS or VxVM from using a cache area.
`sfcache online`	Explicitly makes a cache area available.
`sfcache pin`	Marks a file or directory to be held in the cache until the file or directory is deleted, truncated, or unpinned.
`sfcache purge`	Removes the cached contents for the specified file system.
`sfcache resize`	Resizes the specified cache area.
`sfcache restore-access`	Enables read or write access to files that are missing writeback data. This command does not restore the missing data.
`sfcache rmdev`	Removes the device or devices from use for caching.
`sfcache set`	Sets the values for the specified attributes.
`sfcache stat`	Displays the cache statistics, including cache hit rate, misses, average read and write latencies.
`sfcache unpin`	Removes the file or directory from the pinned state.